T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DNA SIRALARINDAKİ TEKRARLI ÖRÜNTÜLERİN VE POTANSİYEL MOTİFLERİN VERİ MADENCİLİĞİ YÖNTEMİYLE ÇIKARILMASI

Ulaş Baran BALOĞLU

Tez Yöneticisi

Yrd. Doç. Dr. Mehmet KAYA

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ELAZIĞ, 2006

T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DNA SIRALARINDAKİ TEKRARLI ÖRÜNTÜLERİN VE POTANSİYEL MOTİFLERİN VERİ MADENCİLİĞİ YÖNTEMİYLE ÇIKARILMASI

Ulaş Baran BALOĞLU

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Bu tez, ... tarihinde aşağıda belirtilen jüri tarafından oybirliği / oyçokluğu ile başarılı / başarısız olarak değerlendirilmiştir.

Danışman: Yrd. Doç. Dr. Mehmet KAYA
Üye: Yrd. Doç. Dr. İbrahim TÜRKOĞLU
Üye: Yrd. Doç. Dr. Ali KARCI

Bu tezin kabulü, Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ...../...../..... tarih ve ..... sayılı kararıyla onaylanmıştır.

TEŞEKKÜR

Bu tez çalışması sırasında, vatani görevini yapmaktayken bile bana yardımlarını esirgemeyen tez danışmanım Sayın Yrd. Doç. Dr. Mehmet KAYA'ya teşekkürlerimi ve şükranlarımı sunarım.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

SİMGELER VE KISALTMALAR LİSTESİ

A: Adenine;

bp: baz ikilisi;

C: Cytosine;

$C_1 \sim C_m$: Aday motifler;

D: Veri kümesi;

DNA: Deoxyribonükleik asit;

G: Guanine;

GA: Genetik Algoritma;

i: Toplam iterasyon sayısı;

k: Sıra uzunluğu;

max-length: Maksimum motif uzunluğu;

min-length: Minimum altsıra uzunluğu;

n: Sıra sayısı;

p: Örüntü;

RNA: ribonükleik asit;

$S_1 \sim S_n$: Veri kümesine ait sıralar

T: Thymine;

U: Uracil;

Σ: Alfabe;

δ: Minimum destek değeri;

ÖZET

YÜKSEK LİSANS TEZİ

DNA SIRALARINDAKİ TEKRARLI ÖRÜNTÜLERİN VE POTANSİYEL MOTİFLERİN VERİ MADENCİLİĞİ YÖNTEMİYLE ÇIKARILMASI

Ulaş Baran BALOĞLU

Fırat Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

2006, Sayfa: 45

Bu tezde DNA veri kümesinde bulunan biyolojik sıralar üzerinde veri madenciliği yapılarak tekrarlı örüntüler ve potansiyel motifler çıkartılmıştır. Motif bulma problemi olarak adlandırılan bu konuda yapılmış başka çalışmalar da mevcuttur. Fakat çoklu dizi hizalaması kullanan bu çalışmalar performans açısından kötü sonuç vermektedir.

Önerilen yöntem yukarıdan-aşağı veri madenciliği ve genetik algoritma tabanlı hibrit bir çözümdür. Bu yöntemdeki yaklaşım iki temel adımda ele alınabilir. Birinci adım, genetik algoritma kullanılarak aday motiflerin bir popülasyonun oluşturulmasıdır, bunu diğer nesillerin genetik operatörler ve uygunluk fonksiyonu kullanılarak oluşturulması takip eder. İkinci adımda, veri madenciliği yöntemi yukarıdan-aşağı haliyle kullanılarak aday motiflerin uygunluğunun değerlendirilmesi yapılır.

E. coli bakterilerinden alınmış DNA sıralarında önerilen yöntem denenerek uygulanabilirliği ve üstün yanları gösterilmiştir.

Anahtar Kelimeler: Veri madenciliği, biyoenformatik, motif bulma problemi, tekrarlı örüntüler

ABSTRACT

MS Thesis

EXTRACTION OF FREQUENT PATTERNS AND POTENTIAL MOTIFS FROM DNA SEQUENCES WITH DATA MINING METHOD

Ulaş Baran BALOĞLU

Fırat University

Graduate School of Natural and Applied Sciences

Department of Computer Engineering

2006, Page: 45

In this thesis, data mining is applied on DNA datasets in order to extract potential motifs and frequent patterns. There are also other studies on this subject, called motif finding or extraction problem. However, many of those studies suffered from bad performance caused by usage of multiple sequence alignment method.

The proposed method is a hybrid solution, which is based on top-down data mining and genetic algorithm. There are two main motivations of this approach. First, we use genetic algorithm to create a population of candidate motifs, then to create next generations according to usage of fitness function and genetic operators. Second, we use data mining in a top-down manner to evaluate fitness of candidate motifs.

The proposed method was tested on DNA sequences of E.coli bacterias in order to show its superior parts and applicability.

Keywords: Data mining, bioinformatics, motif finding problem, frequent patterns

1. GİRİŞ

DNA ikili sarmal yapısının 1953 yılında James Watson ve Francis Crick tarafından ortaya çıkarılmasından günümüze moleküler biyolojide büyük çapta gelişmeler yaşanmıştır. Bu zaman zarfında büyük çaplı gelişmelerin yaşandığı bir diğer alan olan bilgisayar mühendisliğinin doğal olarak moleküler biyolojiyle yolları kesişmiştir. İki alanın birlikteliğinin bir sonucu olarak biyoenformatik olarak adlandırılan yeni bir alan ortaya çıkmıştır. Her iki alandan birçok araştırmacının yönelim gösterdiği bu yeni alan altında yapılan moleküler biyoloji ve bilgisayar mühendisliği ortak çalışmaları son yılların en çok ilgi çeken konuları arasındadır.

Bu yeni alan altında ortak olarak yapılan çalışmaların ilgilendiği konulardan biri de DNA, RNA ve protein sıralarını içeren biyosıralardaki ilginç örüntülerin ve potansiyel motiflerin keşfidir. Bu örüntülerin biyoloji açısından önemi motiflerin DNA bağlantı noktalarında aldıkları rolden gelir [1]. Bağlantı noktalarının öğrenilmesi, yeni ilaçların bulunmasından doğadaki canlı türleri arasındaki benzerliklerin keşfine kadar birçok konuda yararlı bilgiler sunabileceği için büyük bir öneme sahiptir.

Bu kısımda tez çalışmasının amacıyla beraber yapılan çalışmanın anlaşılmasını kolaylaştırmak amaçlı bazı temel bilgiler verilecektir. İlk olarak biyolojik konularda daha önce bilgi sahibi olmamış okuyuculara ya da hafızasını tazeleme ihtiyacı duyanlara dönük olarak moleküler biyoloji konusu kısaca anlatılacaktır. Bunu takip eden kısımda biyoenformatik alanıyla ilgili temel bilgiler bulunmakta. Tekrarlı örüntülerin ve potansiyel motiflerin bulunması için tezde önerilmiş olan yöntem içerisinde yer alan veri madenciliği ve genetik algoritma konularının anlatılmasından sonra gelen bölümün son kısmında, tezin geri kalanında nelerin bulunduğuna dair bir bakış sunularak birinci bölüm sona erecektir.

1.1. Tezin Amacı

Bu tez çalışmasının amacı DNA verisini girdi olarak alıp bu sıralar üzerinde veri madenciliği yapmaktır. Veri madenciliğinin önerdiğimiz yöntemle uygulanılması sonucunda açığa çıkan bilgi, girdi olarak verilmiş DNA sıralarında gizlenmiş tekrarlı örüntülerin ve potansiyel motiflerin bulunmasını sağlar. Yapılan bu çalışma, farklı alanlardan faydalandığı için çalışmanın daha iyi anlaşılabilmesi, bu alanlar hakkında temel bir bilgiye sahip olunmasını gerektirir.

1.2. Moleküler Biyolojideki Temel Kavramlar

Doğada rastlanılan şeyleri yaşayan ve yaşamayan olarak iki gruba ayırmak mümkündür. Yaşayan şeyler yaşamayanlara kıyasla aktif olarak da tanımlanabilir, çünkü onlar hareket etmek, beslenmek, üremek ve büyümek gibi çeşitli aktivitelerde bulunurlar. Yapılan araştırmalar bugüne kadar maddenin her iki tipte de aynı atomlardan meydana geldiğini ve aynı fiziksel ve kimyasal kurallara uyduğunu ortaya koymuştur. Bu durumda akla gelen ilk soru aradaki bu farkın nedeni olmaktadır. Bu soruya bir cevap ise bilimsel anlamda bugüne kadar verilebilmiş değildir. Bilim, bu cevaplaması zor ve imkansız görünen sorunun yerine canlılarda meydana gelen ve onların yaşamlarını devam ettirebilmelerini sağlayan karmaşık kimyasal reaksiyonlar zincirini anlamaya çalışmıştır [2].

İster karmaşık olsun ister basit, organizmalar benzer bir moleküler kimyaya sahiptir. Yaşayanlar olarak ayrılan grubun yapısında karbohidratlar, yağlar, proteinler ve nükleik asitler gibi çok farklı moleküller bulunur. Ancak yapılan araştırmalar, iki molekülün yaşamın kimyasında temel rol oynadığı bilgisini sunmaktadır: proteinler ve nükleik asitler. Moleküler biyoloji genel bir bakış açısıyla bu iki temel molekülün yani proteinlerin ve nükleik asitlerin fonksiyonlarını ve yapılarını anlamaya çalışır [2].

1.2.1. Proteinler

İnsan vücudunda en fazla bulunan maddelerden biri çok fazla çeşitte bulunabilen proteinlerdir. Vücudumuzda en sık rastlanılan protein çeşitleri yapısal proteinler ve enzimlerdir. Dokular gibi yapım bloklarına yapısal proteinler denirken, kimyasal reaksiyonlarda katalizör görevi yapan proteinler ise enzim olarak adlandırılır [3]. Katalizörler hatırlanacağı üzere kimyasal reaksiyonları hızlandıran maddelerdir.

Proteinler, amino asit olarak adlandırılan daha basit moleküllerin bir araya gelmesiyle oluşur. Proteinlerin yapılarında çoğunlukla 20 farklı aminoasit bulunur. Proteinlerde çok rastlanılan bu aminoasitler, proteinleri anlamak için gereken kimyasal alfabe olarak nitelendirebileceğimiz kısaltmalarıyla birlikte Tablo 1.1 de verilmektedir.

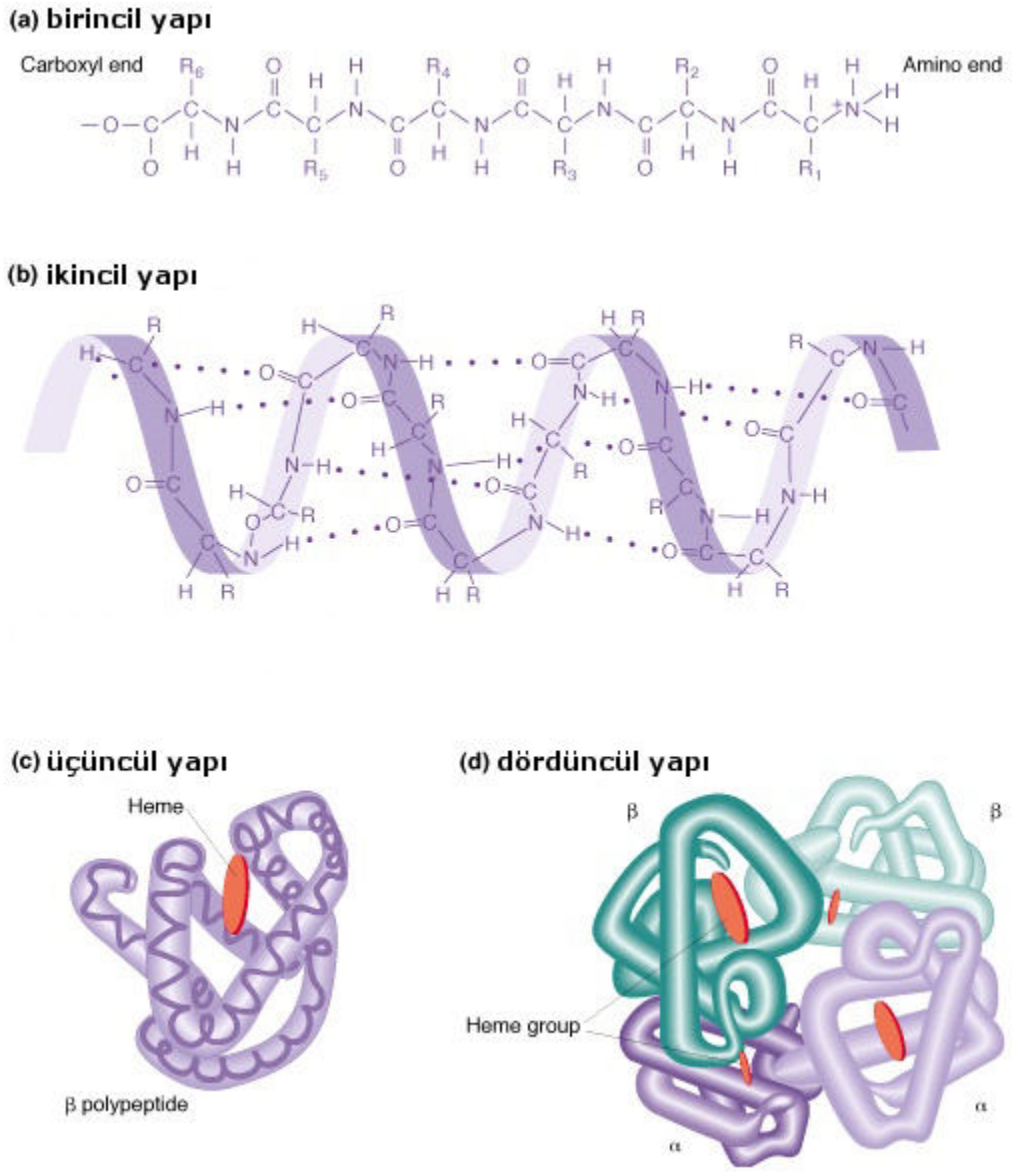
Tablo 1.1. Proteinlerde sıklıkla rastlanılan 20 amino asit.

	Tek-harf Kodu	**Üç-harf Kodu**	**Amino Asit İsmi**
1	A	Ala	Alanine
2	C	Cys	Cysteine
3	D	Asp	Aspartic Acid
4	E	Glu	Glutamic Acid
5	F	Phe	Phenylalanine
6	G	Gly	Glycine
7	H	His	Histidine
8	I	Ile	Isoleucine
9	K	Lys	Lysine
10	L	Leu	Leucine
11	M	Met	Methionine
12	N	Asn	Asparagine
13	P	Pro	Proline
14	Q	Gln	Glutamine
15	R	Arg	Arginine
16	S	Ser	Serine
17	T	Thr	Threonine
18	V	Val	Valine
19	W	Trp	Tryptophan
20	Y	Tyr	Tyrosine

Bir protein oluşurken Tablo 1.1 de isimleri ve kısaltmaları verilen amino asitler bir düzen içerisinde peptide bağlarıyla birbirlerine bağlanırlar. Fakat bu durum proteinleri lineer yapılar olarak düşündürmemelidir çünkü bu birleşme birden çok evreden geçtikten sonra üç boyutlu bir yapının oluşmasıyla sonlanır.

Protein oluşumu sırasındaki evrelerden amino asitlerin birleşmesi birincil yapı (primary structure) olarak adlandırılır. Proteinler üç boyutlu yapılar halinde bulunurlar. Şekil 1.1. de gösterilen ikincil (secondary), üçüncül (tertiary) ve dördüncül (quaternary) yapılar halinde bulunabilirler. Proteinlerin fonksiyonlarının anlaşılması ancak bu üç boyutlu yapıları üzerinden

mümkün olmaktadır. Bu yapının herhangi bir nedenden dolayı hatalı oluşması farklı fonksiyona sahip bir proteinin ortaya çıkmasını sağlar, o farklı yapıdaki protein de bir hastalığa sebep olabilir. Örneğin, son yıllarda ismi sık olarak duyulan deli dana hastalığının ortaya çıkış nedeni bu hatalı protein oluşması durumudur [4].



Şekil 1.1. Proteinlerin birincil, ikincil, üçüncül ve dördüncül yapıları.

Biyoenformatik alanındaki en çok ilgilenilen konulardan bir tanesi de proteinlerin birincil yapılarının elde edilmesinden sonra bu üç boyutlu şekillerinin bulunmasıdır. Çünkü, ancak bu yapı bulunduktan sonra proteinlerin labaratuvar ortamında yapay olarak elde edilmesi mümkün olabilir, bu işlem de hastalıklara tedavi bulunması açısından önemlidir.

Proteinlerin önemi ve fonksiyonları anlaşıldıktan sonra akla gelen ilk sorulardan biri vücudumuzda bulunan proteinlerin nasıl elde edildiğidir. Bu sorunun cevabı hücre içerisinde bulunan ve ribozom olarak isimlendirilen yapılardır. Bir ribozom kendisine gelen messenger ribonükleik asitten aldığı bilgi doğrultusunda amino asitleri teker teker bir araya getirip proteini sentezler. Bu işleyişin nasıl olduğununun anlaşılabilmesi için önce nükleik asitlerin tanınması gerekir.

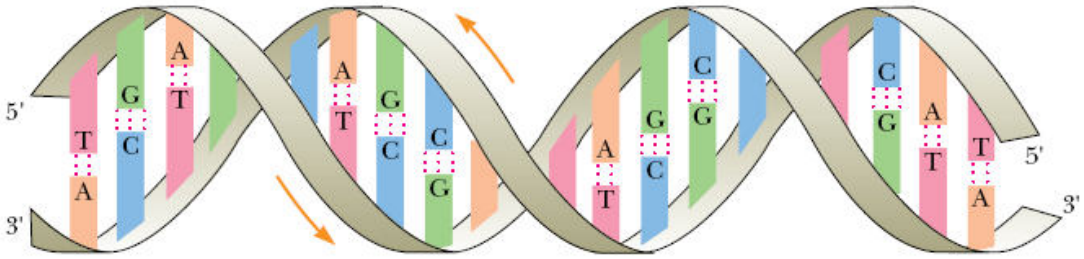
1.2.2 Nükleik Asitler

Canlılarda ribonükleik asit (RNA) ve deoxyribonükleik asit (DNA) olmak üzere iki çeşit nükleik asit bulunur. Bu kısımda sırasıyla DNA ve RNA nükleik asitlerinden bahsedilmektedir.

DNA molekülü de proteinler gibi daha basit moleküllerin zincir halinde bir araya gelmesinden oluşur. Moleküllerin birleşmesi sonucunda DNA molekülüne özgü ikili sarmal olarak adlandırılan bir yapı ortaya çıkar. Bu sarmalın kimyasal yapısına burada değinilmeyecektir ancak bu kimyasal yapı içerisinde yer alan ve baz olarak adlandırılan moleküllerden bahsedilmesinde fayda bulunmaktadır.

DNA sarmal yapısında 4 tip baz bulunur: adenine (A), guanine (G), cytosine (C) ve thymine (T). Bu bazlardan daha büyük yapılara sahip olan A ve G purine olarak adlandırılır; C ve T ise pyrimidine olarak. DNA molekülünün temel yapıtaşı bir bazdan, şeker molekülünden ve fosfat molekülünden meydana gelir ve bu yapı nükleotid olarak isimlendirilir [2]. 200 bazdan meydana gelen bir DNA zinciri ile 200 nükleotidden meydana gelen DNA zinciri denildiği zaman akla gelenler aynı olsa da ikisi birbirine karıştırılmamalıdır.

DNA moleküllerinde bulunan nükleotid sayısı canlıdan canlıya farklılık gösterir. Örneğin, bir insan hücresindeki DNA molekülleri yüzlerce milyon nükleotid ihtiva ederler. Şekil 1.2. de DNA'nın ikili sarmal yapısı ve bazların dizilişi gösterilmiştir.

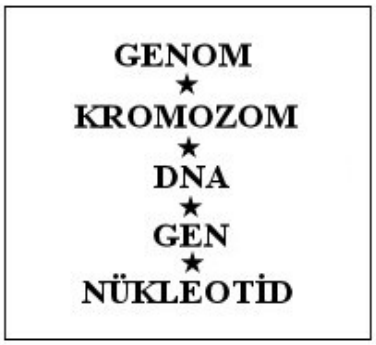


Şekil 1.2. DNA ikili sarmal yapısı.

DNA molekülünün yapısı ikili sarmal olarak tanımlanmıştı. Bu sarmallar ya da zincirler birbirlerini tamamlar bir durumdadır. A bazının karşısına her zaman T bazı gelir, G bazıysa C bazı ile birleşir ve böylece birbirini tamamlayan ikili sarmal yapısı ortaya çıkar. Doğal olarak sarmallardan birisinin yapısı bilindiği zaman diğerinin de baz sırası çıkartılabilir. Bazların bu ikili sarmal yapısındaki gibi bir araya gelmesi sonucu oluşan iki bazın birlikteliği ise baz ikilisi (base pair) olarak adlandırılır, bp olarak kısaltılan bu isimlendirme DNA moleküllerinin bir anlamda ölçü birimidir [2]. 100.000 bp uzunluğunda ya 100 kbp uzunluğunda bir DNA zinciri şeklinde tanımlamalarla sık sık karşılaşılır.

RNA molekülleri de DNA moleküllerine çok benzemektedir. İki molekül arasındaki farklılıklar RNA molekünün deoxyriboz şekeri yerine riboz şekeri taşıması, RNA'da thymine (T) bazı yerine uracil (U) bazının bulunması ve RNA'nın ikili sarmal yapıda olmaması şeklindedir. DNA'nın her zaman tek bir fonksiyon (kodlama) içerip RNA'nın hücre içerisinde farklı fonksiyonlar üstlenmesi de bir farklılık olarak görülebilir.

Şekil 1.3 de canlılarda bulunan genetik materyalin hiyerarşik organizasyonu gösterilmektedir. Haploid bir insan hücresi için bu yapıya nümerik değerler verirsek bu yukarıdan aşağıya hiyerarşik sırasıyla 1 Genom, 23 Kromozom, 23 DNA, ~30.000 Gen, 3 x 10^9 Nükleotid şeklinde olur [5]. Bu organizasyonun şu ana kadar anlatılmayan bileşenleri bir sonraki kısımda anlatılacaktır.



Şekil 1.3. Genetik materyalin hiyerarşik organizasyonu.

1.2.3 Moleküler Genetiğin Mekanizmaları

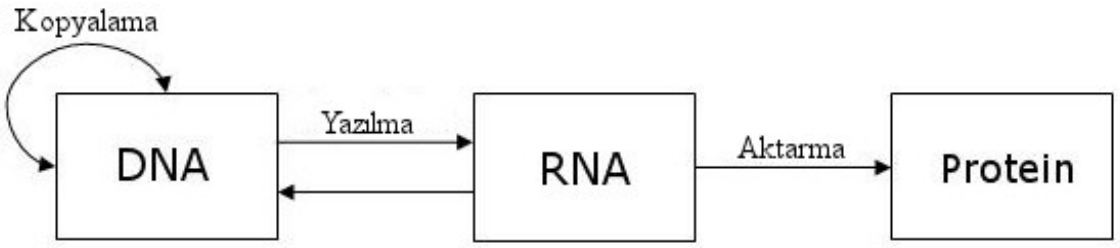
DNA moleküllerini önemli kılan, bu moleküllerin bir organizmada bulunan proteinlerin ve RNA moleküllerinin yapımını sağlayan bilgiyi içermeleridir. Bu bilgi DNA içerisinde kodlanmış halde bulunur.

Bir organizmadaki her hücre birkaç tane uzun DNA moleküne sahiptir. Bu moleküllerin her biri kromozom olarak adlandırılır. DNA hakkında bilinmesi gereken önemli konulardan bir tanesi DNA sıraları içerisindeki belli kısımların proteinlerin yapımı için gereken bilgiyi saklamasıdır, sıranın geri kalanındaysa bu tür bir bilgi bulunmaz. Bilinmesi gereken diğer bir önemli konu ise bir organizmada bulunan her farklı proteinin bilgisinin sadece ve sadece belli bir DNA sırasında bulunmasıdır. İşte bu belli sıralar gen olarak bilinir. Bazı DNA sıralarının RNA oluşumunda rol oynamasından dolayı genler bir protein veya RNA molekünün yapımı için gereken bilgiyi içeren DNA sıraları olarak tanımlanır. Gen uzunlukları değişkendir. Fakat hücre içerisindeki bazı mekanizmalar DNA içerisinde bir genin nerede başlayıp nerede bittiğini anlayabilirler [2].

Protein kısmında anlatıldığı üzere proteinler amino asitlerden oluşmaktaydı. Bu durumda bir proteinin bilgisi, o proteinde bulunan amino asitlerin bilgisini içermelidir sonucuna varılabilir. DNA'nın rolü bu çıkarım yapıldıktan sonra daha iyi anlaşılır. DNA içerisinde bulunan genler bir proteini oluşturan amino asitleri 3'er adet nükleotid kullanarak belirlerler. Bir arada bulunan bu üç nükleotid kodon olarak adlandırılır. DNA kodonları, kodonlar amino asitleri, amino asitlerse proteinleri belirler, bu durum genetik kodlama olarak tanımlanır.

Son olarak DNA içerisinde gizlenen bilgiden proteinlerin elde edilmesinin nasıl olduğuna bakılacaktır. Hücre içerisindeki bazı mekanizmaların genlerin ya da gen dizilerinin DNA içerisinde nerede başladığını anlayabiliyordu. DNA üzerinde promoter adı verilen bölgeler genlerden hemen önce gelerek, o genin rol aldığı hücresel mekanizmayı işaret ederler. Gen ya da gen dizileri bu şekilde tanımlandıktan sonra genin bir kopyası RNA moleküllerine aktarılır. Bu işlemde messenger RNA (mRNA) adı verilen RNA molekülleri rol oynar. Yazılma (transcription) olarak isimlendirilen bu aktarma işlemi sırasında RNA moleküllerinde bulunmayan T bazı U bazıyla değiştirilir.

Bir sonraki aşamada mRNA, protein sentezlemek için hücre içerisinde bulunan ribozomlara hareket eder. Ribozomlarda mRNA molekülleri transfer RNA (tRNA) adı verilen RNA moleküllerine dönüştürülür. Böylece artık genetik koda sahip olmuş olan tRNA molekülü bu kodu aktarma (translation) adı verilen işlemle proteine dönüştürür. Şekil 1.4. de bu anlatılan süreci özetleyen bir gösterim mevcuttur [2].



Şekil 1.4. Hücre içerisinde genetik bilginin akışı.

1.3. Biyoenformatik

1953 yılında DNA yapısının çözümlenmesinden bugüne moleküler biyoloji inanılmaz bir gelişim gösterdi. Biyomoleküler dizilerde değişiklik yapma yeteneğinin kazanılmasından sonra büyük miktarlarda veri yaratıldı ve halen yaratılmaya devam etmekte. Labaratuvarlarda açığa çıkan bu büyük çaplı verinin incelenmesi zaman içerisinde disiplinler arası yeni bir alanın ortaya çıkmasına neden olmuştur. Biyolojik bilimlerle ilgilenen bilim adamları bu verileri bulan ve en üst seviyede kullanan kişilerdir. Fakat bu verilerin çok oluşu ve karmaşıklığı matematik ve bilgisayar bilimleri gibi diğer alanlardan yardım alma ihtiyacını doğurmuştur. Bu ihtiyaçlar ve

ortak çalışma sonucu doğan yeni alan biyoloji ve enformatik kelimelerinin birleştirilmesiyle elde edilen biyoenformatik kelimesiyle isimlendirilmiştir [2].

En geniş haliyle moleküler biyolojideki problemlerin çözümünde matematiğin ve bilgisayar bilimlerinin kullanılması olarak ifade edilir. Veritabanları bilindiği gibi elde edilen bilginin saklanmasında kullanılmaktadır. Moleküler biyoloji kullanılarak elde edilen diziler de diğer veriler gibi veritabanlarında saklanır. Bu veriler depolanmaya başladıktan sonra bilim adamları moleküler biyolojinin kullanımına has veritabanı modellerine ihtiyaç duymuşlardır. Çünkü mevcut modellerin, moleküler dizilerin anlaşılması gibi konulara çok olanak sağlamadığı farkedilmişti. Bu dizilerin anlaşılması için yeni veritabanı modelleri ortaya konulmalıydı ve bu veritabanlarında örüntü analizi, dizi hizalaması gibi işlemleri yapabilecek bir takım bilgisayar programları yazılmalıydı. Bu da zaman içerisinde yukarda bahsedilen alanlar arası işbirliğini doğurmuştur.

1.3.1. Biyoenformatiğin Tanımı

Kelime kökeni olarak incelendiğinde biyoloji ve enformatik kelimelerinin birleşmesi sonucu ortaya çıkmış bir kelimedir. Enformatik kelimesi ‘information’ yani bilgi kelimesine Yunanca’daki ‘-tic’ takısının getirilmesi sonucu elde edilmiş bir kelimedir ve ‘teori’ anlamına gelmektedir. O halde kaba bir biçimde tanımlanacak olursa moleküler biyolojideki problemlerin çözülmesinde bilgisayar bilimine ait tekniklerinin kullanılması ve geliştirilmesi denilebilir.

En sade haliyle biyoenformatik biyolojik sorular sormak ve bu sorulara cevap bulmak için teknolojinin kullanılmasıdır. Biyoloji terim olarak kullanılsa da dikkat edilmesi gereken bu alanın moleküler biyoloji ile ilgileniyor olmasıdır. Hesapsal Biyoloji (Computational Biology) ve Biyocomputing benzer anlamlar taşıdığı için biyoenformatikle karıştırılabilir. Biyoenformatiğin bu alanlardan temel farkı daha çok bilgiyi keşfetmeye dayanan algoritmalar kullanmasıdır.

1.3.2. İnsan Genomu Projesi

İnsan Genomu Projesi 1988 yılında başlamış olan bir çok ülkenin katıldığı bir araştırmadır. Bu projedeki amaç insan kromozomlarının ve DNA dizisinin bütün olarak fiziksel bir haritasını çıkarmaktır. Bu projenin bir parçası olarak doğadaki bakteri, fare, sinek gibi diğer organizmaların da genomları incelenmeye alınmıştır.

Bu çalışmaya insan dışındaki diğer canlıların neden dahil edildiği sorusu sorulabilir. Burada amaçlanan insan genomunun karmaşık oluşu nedeniyle daha basit genomlar üzerinde araçların geliştirilmesi ve sonra insan genomunda denenmesidir. Ayrıca sadece insan değil bütün canlı türleri moleküler biyolojinin inceleme alanı içerisindedir.

İnsana ve diğer canlılara ait bu dizilerin tamamı bütün olarak çıkartıldıktan sonra daha zor olan analiz etme işlemine sıra gelecektir. Bu diziler arasındaki analiz genetik hastalıkların tedavisi için ilaç bulunmasından türler arasındaki akrabalıkların keşfine kadar bir çok farklı araştırmaya imkan sağlayacaktır [2].

1.3.3. Dizi Veritabanları

Geçmiş yıllarda büyük miktarlarda DNA, RNA ve protein dizileri keşfedildi ve bu veriler kimi kurumlar tarafından sağlanan veritabanlarında saklandı. Biyolojik veriler üzerinde veri madenciliği yapacak araştırmacılara kaynak sağlayacak bu veritabanlarından bu kısımda bahsedilmektedir [2].

GenBank: Amerika Birleşik Devletleri'nde National Center for Biotechnology Information (NCBI) tarafından yüzbinlerce DNA sırasının bulunduğu bir veritabanıdır [6]. Türlere göre gruplandırmalar yapılmıştır. Örneğin "PLN" bitkilerde bulunan sıraları ifade eder, "BCT" ise bakterilerde bulunanları. Bu veritabanı üzerinde araştırmalar anahtar kelimelerle veya dizilerle yapılabilir.

EMBL: The European Molecular Biology Laboratory farklı dizi ambarlarını saklayan bir enstitüdür [7]. Verilerin organize ediliş biçimi GenBank'a benzese de verilerin alan adları ve tanımlanma kodlamaları oldukça farklıdır.

GDB: The GDB Human Genome Database insan genomuna ait segmentlerin ve diğer biyolojik verilerin saklandığı bir veritabanıdır [8].

RCSB PDB: The Protein Data Bank proteinlerin 3-boyutlu yapılarının saklandığı bir veri ambarıdır [9].

1.4. Veri Madenciliği

Ünlü düşünür Francis Bacon "Bilgi güçtür" der. Tarihsel anlamda geriye bakıldığında güçlü olan toplumların tamamına yakınının, yaşadıkları dönemdeki diğer toplumlara göre daha

fazla bilgiye sahip olduklarını görülür. Gücü getiren bu bilgi askeri anlamda savaş stratejilerinde, bilimsel anlamda yeni keşiflerin yapılmasında ve başka konularda da görülebilir.

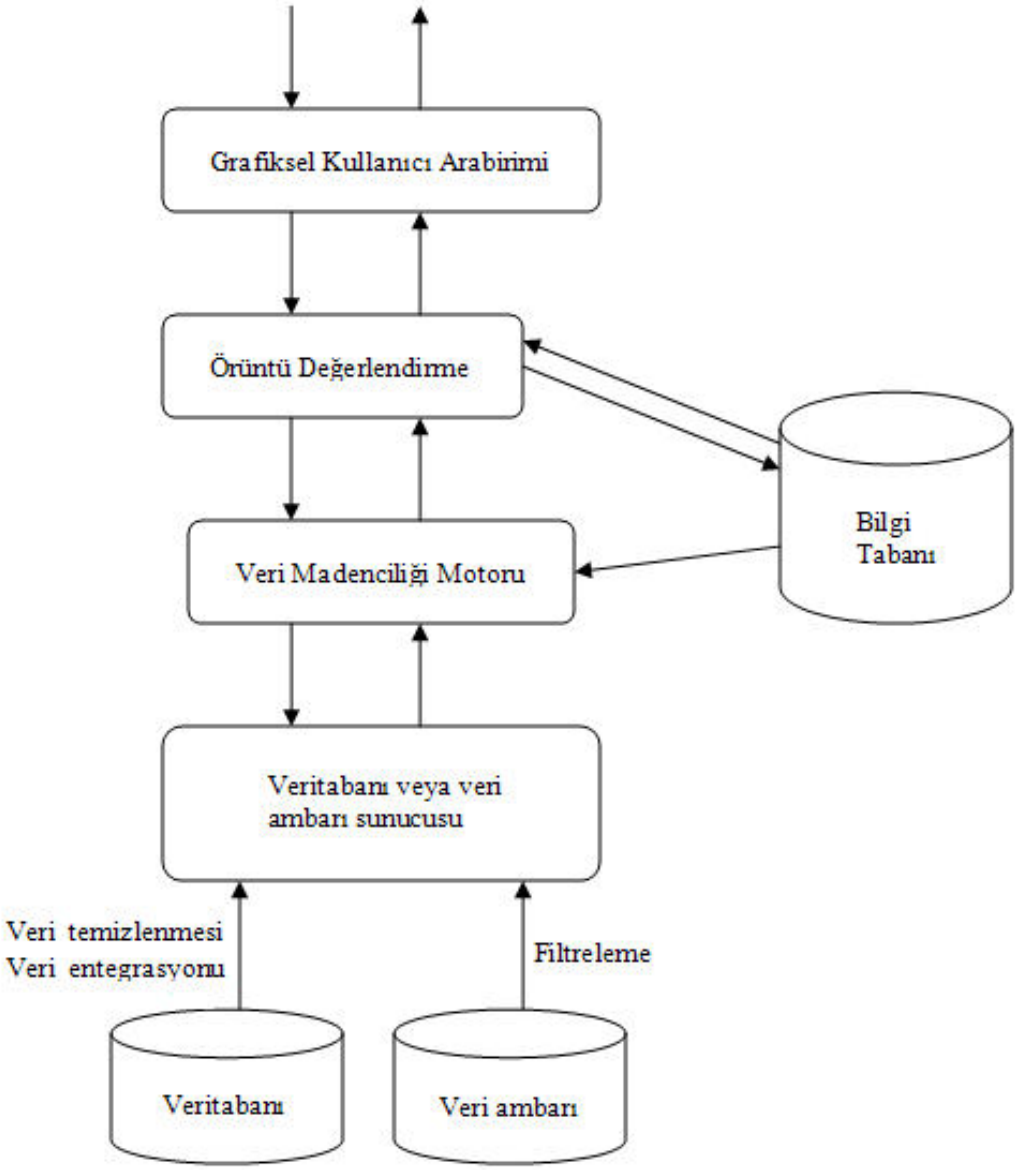
Son yıllarda artan bilgisayar kullanımı, çeşitliliği artan yazılımlar ve internet, elektronik ortam olarak adlandırılan platformda yüksek miktarlarda verinin depolanmasına neden olmuştur. Bu verilerden faydalı bir bilgi ya da Bacon'ın bakış açısıyla güç elde etme düşüncesi veri madenciliğinin ortaya çıkmasına neden olmuştur. Kısaca veri madenciliğinin bilişimdeki hızlı gelişmenin kaçınılmaz bir sonucu olarak ortaya çıktığı söylenebilir.

Veri madenciliği fikrinin ortaya çıkışı daha teknik bir biçimde ele alınacak olursa, bu 1960'lardan günümüze veritabanlarının ve bilişim teknolojilerinin ilkel dosya işleme sistemlerinden gelişmiş ve güçlü veritabanı sistemlerine doğru evrimleşmesine bağlanabilir. Çok hızlı artan ve artık büyük miktarlara ulaşan bu veriler, dağıtık bir biçimde büyük veritabanlarında toplanılmaya başlanmıştır. Büyük miktarlardaki bu veri yığınlarını, bir insanın herhangi bir araç kullanmadan anlayabilmesi pek mümkün değildir. Veriler ve içerdikleri bilgi arasındaki boşluğu kapama ihtiyacı veri madenciliği araçlarının geliştirilmesine neden olmuştur [10].

1.4.1 Veri Madenciliğinin Tanımı

Bir tanım yapmaya çalışılırsa en basit biçimde veri madenciliği büyük miktarlardaki veriden bilgi çıkarma olarak tanımlanabilir. Daha kapsamlı bir tanım yapılacak olursa, veri madenciliği bir veya daha fazla makine öğrenme tekniğinin uygulanarak otomatik olarak bir veritabanı içinde bulunan verilerden bilgi çıkartılması, verilerin analiz edilmesi işlemidir. Topraktan altın çıkarılmasına toprak madenciliği değil de altın madenciliği denildiği düşünülürse, veri madenciliği yerine bilgi madenciliği ifadesini kullanalım şeklinde düşünülebilir. Fakat bu ifadenin İngilizce karşılığı olan "knowledge mining from data" ifadesi pratik kullanıma ters düşecek biçimde uzun bir ifade olarak değerlendirilmiştir ve "data mining" ifadesinin yapılan işi daha iyi açıkladığı düşünülmüştür. Bu iki nedenden ötürü de yapılan iş veri madenciliği olarak adlandırılmıştır [10].

Veri madenciliğiyle aynı anlama gelen ya da yakın bir anlam içeren bazı başka tanımlamalarla da literatürde karşılaşılabilinir. Bunlar veritabanlarında bilgi madenciliği, bilgi çıkarımı, veri/örüntü analizi, veri arkeolojisi ve veri taranmasıdır.



Şekil 1.5. Tipik bir veri madenciliği sisteminin mimari yapısı

Şekil 1.5. de bir veri madenciliği sisteminin mimari yapısı gösterilmektedir. Bu yapıda aşağıdaki temel modüller bulunmaktadır [10]:

- **Veritabanı, veri ambarı veya başka bir bilgi deposu**: Bu bir ya da birden fazla veritabanından veya başka bilgi kaynaklarından oluşabilir. Bu kaynaklardaki veri üzerinde veri temizlemesi ve bu kaynaklara başka kaynaklardan veri entagrasyonu yapılır.
- **Veritabanı veya veri ambarı sunucusu**: Kullanıcıdan gelen veri madenciliği işlemine göre ilgili veriyi getirmekten sorumlu sunucudur.
- **Bilgi Tabanı**: Araştırmayı yönlendirecek veya çıkan örüntülerin ilgi derecelerini tespit edecek bilgi kaynağıdır.
- **Veri madenciliği motoru**: Bu modül veri madenciliği sistemlerinin olmazsa olmaz kısmıdır. İdeal olanında ilişkilendirme, sınıflandırma, cluster analizi, türev analizi, karakterizasyon gibi fonksiyonel modüller bulunur.
- **Örüntü değerlendirme modülü**: Bu modülün genel olarak görevi ilgi boyutunu ölçmektir ve ilgi çeken örüntüler üzerinde arama yapmak için diğer veri madenciliği modülleriyle etkileşimde bulunmaktır. Bu kısım veri madenciliği motoruna kısmına da katılabilir.
- **Grafiksel kullanıcı arabirimi**: Bu modül kullanıcılar ve veri madenciliği sistemi arasındaki haberleşmeyi sağlar, kullanıcının bir veri madenciliği sorgusu ya da görevi yaptırabilmesi için gereken etkileşimleri bulundurur.

1.4.2. Mevcut Problemin Veri Madenciliğine Uygunluğu

Bir problemin çözümünde veri madenciliği kullanmanın uygun olup olmadığını kestirebilmek zor bir iştir. Başlangıç noktası olarak dört genel soru düşünülebilir [11]:

1. Problemi açık bir biçimde tanımlayabiliyor muyuz?
2. Potansiyel olarak anlamlı bir veri mevcut mu?
3. Veriler gizli bir bilgi içeriyor mu ya da veriler nedensel olup sadece raporlama amaçları için mi uygun?
4. Veriyi işlemenin maliyeti veri madenciliği projesi sonucu elde edilen potansiyel bilginin getirisine kıyasla daha karlı mı?

Bu sorulara cevap ararken çıkartılacak bilginin aşağıdaki biçimde önceden sınıflandırılması probleme veri madenciliği uygulama konusunda karar vermeyi kolaylaştırır [11].

- **Yüzeysel Bilgi**: Doğal olarak nedenseldir. Bu tip bilgi veritabanlarında kolayca depolanır ve manipüle edilir. SQL gibi veritabanı sorgulama dilleri veriden yüzeysel bilgi çıkarmak için mükemmel araçlardır.
- **Çokboyutlu Bilgi**: Bu tip bilgi de nedenseldir, fakat bu tip bilgide veri çokboyutlu biçimde depolanmıştır. Çevrimiçi Analitik İşleme (OLAP) araçları çokboyutlu veri üzerinde kullanılan araçlardır.
- **Gizli Bilgi**: SQL gibi sorgulama dilleri kullanılarak kolayca çıkartılamayan verideki örüntüleri veya düzenlilikleri temsil eder. Ancak, veri madenciliği algoritmaları bu tip örüntüleri kolayca bulabilir.
- **Derin Bilgi**: Veritabanında depolanmış olup sadece ne aradığımız konusunda bir yön verirsek bulabileceğimiz bilgi tipidir. Günümüzde mevcut olan veri madenciliği araçları bu tip bilgiyi bulma yeteneğine sahip değildirler.

1.4.3. Veri Madenciliği Metodları

Verilerden örüntüleri çıkarma şeklinde bir yol izleyen veri madenciliği süreci genel olarak sınıflandırma, regression analizi, link analizi, bölümleme veya sapma tespiti gibi metodları da kullanır [12].

Sınıflandırma verinin daha önceden tanımlanmış ya da yeni keşfedilmiş sınıflara eşlenmesini içerir. İlk durumda, önceden tanımlanmış örneklerden oluşan bir küme veritabanı içerisinden seçilip alınmış verilerin sınıflandırmasında kullanılacak bir model geliştirir. Sonraki durumda, sistem verinin analizine bağlı olarak verileri sınıflandırmak için kendi modellerini geliştirir.

Regression analizine dayanan veri madenciliği, veriye istatiksel metodlara dayanarak sürekli nümerik bir değişken atanmasını içerir. Bunu uygulamadaki amaçlardan bir tanesi bir kaç veri örneğinden eğilimlerin tahminidir.

Bağlantı analizi veritabanındaki veriler arasında kolayca gözüken bağlantıları değerlendirir. Bu analiz veri içerisinde bağlantı oluşturabilecek bağıntıları (korelasyonları) vurgular fakat neden oluştuklarıyla ilgili bir incelemede bulunmaz.

Sapma tespiti, mevcut modellerde tanımlanmış veya gözlemler sonucu ortaya çıkarılmış belli bir standardın dışında kalan verileri tespit etmek için kullanılır.

Bölümlemeye dayanan veri madenciliği veri sınıflarını veya gruplarını belli bir metriğe göre benzer davranır şeklinde tanımlar. Bölümleme bağlantı analizine benzemektedir, farkı veri üzerinde noktalar yerine gruplar üzerinde uygulanmasıdır.

Veri madenciliğinin bu metodları genelde paralel olarak ya da ardışık bir işlemin parçaları olarak bir arada kullanılmaktadır. Örneğin, bölümleme sınıfların bir sınıflandırma işlemi üzerinden belirlenmesine ihtiyaç duyar. Benzer biçimde, bağlantı analizi istatiksel bir analiz üzerinden şekillenir, korelasyon katsayılarının mevcut olmasına ihtiyaç duyar. Sapma tespiti verinin uygun bir biçimde sınıflandırılmış olduğunu varsayar ve normal modeli tanımlarken istatiksel bir değerlendirme yapar.

1.4.4. Bağıntı Kuralları

Çok rastlanılan veri madenciliği uygulamalarından bir tanesi market sepeti analizi olarak bilinen bir alışveriş sırasında müşterinin birlikte alma eğilimi gösterdiği ürünlerin saptanmasıdır. Bu analizin sonrasında müşteri-alışveriş davranışı ortaya çıkarılır. Bir bağıntı kuralı bu davranışı ortaya çıkaran kuralları uygun biçimde göstermeye yarar. Örneğin, "süt alan bir müşteri yumurta da satın alır" muhtemel bir bağıntıdır. Bu bağıntının bir kurala dönüşebilmesi için destek değeri ve güven değeri olarak tanımlanan iki şartı sağlaması gerekir.

Bir X nesnesi ya da ürünü için destek değeri, X içeren veritabanı kayıtlarının bütün kayıtlara oranıdır. Destek değeri tekrarlanan örüntülerin frekanslarıdır da denilebilir. Güven değeri ise bir muhtemel bağıntı kuralının gücünü ölçmeye yarar. Bu iki kavram matematiksel gösterimle aşağıdaki gibi ifade edilir [13]:

- Bir $X \rightarrow Y$ kuralının destek değeri, $X \cup Y$'nin destek değeridir.
- Bir $X \rightarrow Y$ kuralının güven değeri, $X \cup Y$'nin destek değerinin X'e ait destek değerine oranıdır.

1.4.5. Veri Madenciliği Araçları

Uygulamaların çoğu için, veri madenciliği bir genetik algoritma yazılmasına ya da yapay sinir ağı tasarlanılmasına ihtiyaç duymaz. Bunların yerine, çeşitli yüksek seviyeli programlama

dillerini kullanır ya da genel amaçlı veya biyoenformatik amaçlı diğer bazı araçları kullanır. Tablo 1.2. de yaygın kullanılan araçlardan bazıları verilmiştir [12].

Tablo 1.2. Veri Madenciliği İçin Kullanılan Araç Çeşitleri ve Örnekleri.

Kullanılan Araç	**Örnekler**
Programlama Dili	Perl, Python, SQL, XML
Genel-Amaçlı	Angoss, Clustran, Cross-Graph, Cross-z, Daisy, Data Distilleries, Database Marksman, DataMind, GVA, IBM Intelligent, Miner, Insightful Miner, Integral Solutions, KXEN, Magnify, MatLab, NeoVista Solutions, Oracle Darwin, Quadstone, SAS, Spotfire, SPSS Clementine, StatPac, Syllogic, ThinkAnalytics, Thinking Machines, Weka
Biyoenformatik Amaçlı	MEME, PIMA, Pratt, PrattWWW, SPEXS

Veri madenciliğini biyolojik veriler üzerinde kullanırken en çok kullanılan programlama dilleri Perl, Python, ve SQL'dir. Perl ve Python script yazma dilleridir ve metine dayalı veya sıralı veriler üzerinde madencilik yaparken oldukça kullanışlıdırlar. Bu diller esnek bir yapıya sahiptirler ve bir çok ihtiyacı gerçekleme yeteneğine sahip olmalarından dolayı yetenekli dillerdir. Ancak Perl ve Python'un script dili olmalarından kaynaklanan bir kısıtlaması mevcutcur. C++ ve diğer programlama dillerinden farklı olarak scriptler derlenmezler bunun yerine bir yorumlayıcı tarafından runtime esnasından çalıştırılırlar. Bunun sonucu olarak, Python veya Perl kullanılarak yapılan programlar aynı algoritmayı kullanan C++ programlarına göre daha yavaştırlar fakat daha kolay yazılırlar ve kodları daha anlaşılır olur. İki dil arasında yavaş çalışmayı karşılaştıracak olursak, Python'la yazılmış kodların çalışma hızı Perl'le yazılmış olanlara göre daha hızlıdır. Bunun da nedeni bu dilin C++ ile yazılmış modüllerden temellenmiş olmasıdır. Üzerinde durulması gereken bir nokta da bu dillerin derlenmeden çalıştırılmasının bu bahsettiğimiz hız konusundaki dezavantajı yerine göre ortadan kaldırabilmesidir. Derlenme zamanı olmadığı için burdan elde edilen zaman, bu kodların yavaşlığından kaybedilen zamanı telafi edebilmektedir. Son olarak, Python ve Perl'in bir diğer avantajı bu dillerin açık-kaynaklı, özgür yazılımlar olmasıdır [12].

Yukarda bahsedilen diller gibi SQL'de yorumlayıcı kullanan bir dildir. Ancak, SQL Perl veya Python gibi esnek bir dil değildir, sadece ilişkisel olan bir veritabanını sorgulamada kullanılabilir. Bunun faydası ise yüksek performans olarak ortaya çıkar. Ek olarak belirtilmesi gereken bir durum SQL'in tek başına bir uygulama olmamasıdır. Genel olarak bir üreticinin veritabanı yönetim sistemi yazılımının bir parçasıdır. SQL kullanmanın avantajı bu dilin farklı firmalara ait veritabanı yönetim sistemleri arasında taşınabilir olmasıdır, bu sayede araştırma yapan bir kişi farklı veritabanı sistemlerini yeni bir sorgulama dili öğrenmeye ihtiyaç duymadan sorgulayabilir. Veritabanı ister Oracle, ister Microsoft, ister IBM tarafından üretilsin, bundan bağımsız olarak SQL komutları aynı kalır. SQL komutları veritabanı üzerinden girilebilirler, fakat genel olarak bir başka programlama dili içine gömülüp kullanılırlar, bu sayede o programlama dili veritabanı üzerinde çeşitli işlemler yapma yeteneği kazanır [12].

XML bir veri biçimleme dilidir. Genişletilebilirliği ve veri madenciliğine dönük kullanımı olan etiket kullanımı özellikleri sayesinde çevrimiçi veritabanı uygulamalarında günümüzde popülerdir. XML ile inşa edilmiş bir veritabanı ya da veri kümesi sadece SQL'i ve standart ilişkilendirme tablolarını destekleyenlere göre veri madenciliğine daha hazır durumdadır. Bütün avantajlarına karşın XML kullanımının yarattığı dezavantajlar da mevcuttur. Dilin esnekliği nasıl genişletileceğini sınırlandırmadığı için, farklı programcılar tarafından XML kullanılarak yazılan veritabanları birbirlerine çok az benzerlik gösterir [12].

Programlama dillerine ek olarak, veri madenciliği amaçlı kullanılabilecek yüzlerce araç mevcuttur. Tablo 1.2. de bu araçların popüler olanlarının bir kısmı verilmekte. Bu araçlardan Oracle Darwin gibi bazıları belli bir veritabanı ürününde çalışacak şekildeyken SAS gibi olanları ise herhangi bir veritabanında kullanılabilir. Bu uygulamalardan gene bazıları, örneğin MatLab çok çeşitli veri madenciliği yeteneklerine sahiptir. MatLab tabloda verilen araçlar arasında oldukça kullanışlı bir örnektir. Bunun nedeni performansın en çok önem verilen konu olmadığı durumlarda, bir araştırmacı Perl veya Python, veya SQL bilgisini MatLab bilgisiyle birleştirip veri madenciliği ile ilgili bir çok zor problemi çözebilir [12].

Tablo 1.2. de ayrıca biyoenformatik amaçlı veri madenciliği araçlarından örneklerde verilmekte. Bunlar MEME, Pratt, PIMA, ve SPEXS'dir. MEME (Multiple Em for Motif Elicitation) motif bulan bir araçtır. Pratt ise tek başına bir örüntü bulma aracıdır ve hizalanmamış protein sıralarında saklı örüntüleri açığa çıkarmaya yarar. Kullanıcı çeşitli parametreler girerek nasıl örüntüler aranacağını, bir örüntüde ne kadar sıra olacağını belirleyerek programı kullanır. PrattWWW, Pratt'ın web üzerinden kullanılabilir halidir ve bir Java appletiyle kullanıcıya arabirim sağlar. PIMA (Pattern-Induced Multi-sequence Alignment program) bir dizi kümesinde çoklu-dizi hizalama yapabilen bir programdır. SPEXS (Sequence Pattern EXhaustive Search) bir

sıralı örüntü bulma aracıdır. Sadece biyolojik veriler için optimize edilmiş bu veri madenciliği araçlarının en önemli özelliği hem performans hem de buldukları kurallar bakımından çok etkili olmalarıdır. Bu araçların kötü yanıysa yapılacak veri madenciliği farklı bir biyolojik yapıya kayarsa yetersiz kalmalarıdır, örneğin yapılan madencilik nükleotid dizilerinden çıkıp protein yapılarına kayarsa bu araçlar için yeni paketler kullanılmasına ihtiyaç duyulur ya da yeni araçlar kullanmak gerekir [12].

1.5. Genetik Algoritma

Genetik Algoritma (GA) fikri 1960 yılında John Holland tarafından bulunmuştur. Michigan Üniversitesinde Holland, öğrencileri ve çalışma arkadaşlarıyla 1960-70 yılları arasında bu algoritmayı geliştirmiştir. Evrimsel stratejilerden ve evrimsel hesaplamadan farklı olarak Holland'ın hedeflediği şey belli bazı problemleri çözmek için algoritma geliştirmek değildi, bunun yerine doğada meydana gelen adaptasyon fenomeni üzerinde çalışmak ve bu doğal adaptasyonun mekanizmasını bilgisayar sistemlerine aktarabilmek için yöntem geliştirmekti.

Genetik algoritma için bir tanımlama yapmaya çalışırsak, onlara arama metodları doğal bir fenomeni modelleyen stokastik algoritmalar diyebiliriz. Michalewicz [14] genetik algoritmaların arkasındaki fikrin doğa ne yapıyorsa yap olduğunu iddia eder. Bu iddiasını da bir örnekle somutlaştırır.

Örnek olarak tavşanları alalım, herhangi bir zamanda bize verilen bir tavşan popülasyonu olsun. Bu tavşanlardan bazıları diğerlerinden daha hızlı ve zekidir. Bu hızlı ve akıllı tavşanların doğal olarak diğerlerine göre tilkilere daha az yem olması beklenir. Yavaş ve daha aptal olan tavşanların da tabi ki tamamı yem olmaz, bazıları şansları yardımıyla hayatta kalır. Bu tavşan popülasyonunda üreme düşünülürse daha fazla sayıda zeki ve hızlı tavşan hayatta kalacağı için onların içinde çok bulunacağı bir üreme gerçekleşir. Sonuç olarak, yeni popülasyon eskisinden zeki ve hızlı olanları bulundurması ve yeni gelen bireylerin de zeki ve hızlı olma olasılığının artması sonucunda bir önceki popülasyona hız ve zeka ortalamaları açısından üstünlük sağlar. Bir genetik algoritma da adım adım tavşanların hikayesine benzer bir prosedür takip eder [14].

Genetik algoritmaların bir çoğunda bazı parçalar ortak bir biçimde bulunmaktadır. Bunlar uygunluk fonksiyonuna bağlı seçme işlemi, yeni bireyleri oluşturmak için çaprazlama (crossover) ve mutasyondur. Holland, inversion isimli dördüncü bir parça daha kullanmıştır ancak günümüzde çoğu genetik algoritma inversion içermez. Belli bir problem için düşünülen genetik algoritmanın aşağıda verilen beş bileşene sahip olması gerekir [15]:

• Problemin olası sonuçları için bir genetik reprezantasyona
• Potansiyel çözümlerin başlangıç popülasyonunu oluşturmak için bir metoda
• Çözümleri değerlendirme için bir uygunluk fonksiyonuna
• Bireyler üzerinde değişiklikler yapabilecek genetik operatörlere
• Genetik algoritmanın kullandığı bazı parametreler (popülasyon büyüklüğü, genetik operatörlerin uygulanma olasılılıkları, vs.) için başlangıç değerlerine

Bu bölümün diğer kısımlarında kısaca bu bileşenlere değinilmektedir.

1.5.1. Uygunluk Fonksiyonu

Genetik algoritmalar genellikle mevcut popülasyondaki bireylere bir uygunluk değeri (score, fitness value) atayan bir uygunluk fonksiyonuna (fitness function) ihtiyaç duyarlar. Bu fonksiyon belirlenirken problemin nasıl daha iyi çözümleneceği düşüncesinden yola çıkılır. Aşağıda örnekte bir uygunluk fonksiyonu verilmekte:

$$f(y) = y + 2|\sin(16y)|,\ 0 \leq y < \pi$$

Bu uygunluk fonksiyonu belli bir problemin çözümüne gayet iyi uyarken başka bir problemde uygulanılması durumunda iyi bir sonuç vermeyebilir. Başka bir ifadeyle uygunluk fonksiyonları genel olarak probleme özeldir.

1.5.2. Genetik Algoritma Operatörleri

En temel genetik algoritma üç tip operatör içerir: seçme (selection), çaprazlama (crossover) ve mutasyon. Bu operatörler aşağıda verilen fonksiyonları yerine getirirler:

- **Seçme**: Bu operatör popülasyondan üreme için bireyleri seçer. Bireyin uygunluk fonksiyonu ne kadar istenen düzeydeyse üreme için seçilme şansı ve sayısı o kadar fazla olur.
- **Çaprazlama**: Bu operatör rasgele olarak bir yer belirler ve belirlediği bu yerin sağında ve solunda kalan parçaları iki kromozom arasında değiştirerek iki yeni birey meydana getirir. Örneğin, 10000100 ve 11111111 zincirleri üçüncü karakterden hemen sonra

çaprazlama olacak biçimde birleştirilirse meydana gelen yeni bireyler 10011111 ve 11100100 zincirleri olur.

- **Mutasyon**: Bu operatör bir kromozomda rasgele değişiklik yapar. Örneğin, 00000100 zincirine mutasyon operatörü uygulanırsa ve mutasyona uğrayan kısım baştan ikinci karakter olursa zincir 01000100 halini alır. Mutasyonlar düşük birer olasılıkla oluşurlar, meydana gelme nedenleri popülasyonda genetik çeşitlilik yaratmaktır.

1.5.3. Basit Bir Genetik Algoritma Örneği

Açık bir biçimde tanımlanmış bir problem olsun, bu problemin çözümünde kullanılabilecek genetik algoritma aşağıdaki gibi çalışır [15]:

1. Rasgele oluşturulmuş n tane kromozoma (probleme aday çözüm) sahip popülasyon ile başla.
2. Popülasyondaki her bir kromozom x için uygunluk değeri *f*(x)'i hesapla.
3. *n* tane yeni birey oluşturulana kadar aşağıdaki adımları tekrarla:
a. Mevcut popülasyondan uygunluk değeri çok olanlara daha çok seçilme olasılığı verecek şekilde kromozomlar seç. Seçme işlemi sırasında bir kromozom birden çok defa seçilebilir olsun.
b. Belli bir olasılıkla (çaprazlama olasılığı) rasgele seçilmiş noktalara çaprazlama operatörü uygula.
c. Belli bir olasılıkla bireyler üzerinde mutasyon operatörünü uygula.
4. Mevcut popülasyonu yeni oluşturulmuş popülasyonla değiştir.
5. Adım 2'ye dön.

Bu sürecin kaç defa tekrar edileceği bir parametre olarak algoritmaya girilir. Her bir tekrar yeni bir jenerasyon olarak adlandırılır. Tipik bir genetik algoritma 50 ile 500 arasında veya daha fazla sayıda jenerasyon içerir. Jenerasyonların tamamı bittikten sonra yapılan işleme algoritmanın çalıştırılması denir. Algoritma çalıştırıldıktan sonra popülasyonda bir veya daha fazla uygunluk değeri çok yüksek kromozom bulunur. Her bir çalıştırmada rasgelelik işin içine gireceği için farklı sonuçlar elde edilir. Bu nedenle genetik algoritma kullanan araştırmacılar algoritmalarıyla beraber kullandıkları parametreleri ve bazı istatistiksel bilgileri de verirler. Burada verilen basit genetik algoritma örneği ufak tefek değişikliklerle birçok yerde kullanılır.

1.6. Teze Bakış

Tez çalışmasının ikinci bölümünde; yukarıdan-aşağı veri madenciliği konusu verilecek, üçüncü bölümde; motif arama problemi tanımlanarak MEME, Gibbs Sampler gibi popüler motif bulma algoritmaları sunulmuştur, dördüncü bölümde; motif arama problemine önerilen çözüm yöntemi gerekli tanımlamalar ve algoritması verilerek anlatılacak ve bölümün sonunda elde edilen deneysel sonuçlar tartışılarak verilecektir, beşinci bölümde; Top-Down GA olarak adlandırılan dördüncü kısımda önerilen yöntemi gerçekleyerek motif arama işlemi yapan program tanıtılacaktır; altıncı bölümde; tez çalışması süresince elde ettiğimiz sonuçlar ve ileriye dönük düşünceler sunulmaktadır.

2. YUKARIDAN-AŞAĞI VERİ MADENCİLİĞİ YÖNTEMİ

2.1. Giriş

Biyoenformatik kısmında da bahsedildiği üzere son yıllarda GenBank [16] ve SWISS-PROT [17] gibi içlerinde büyük miktarlarda DNA ve protein dizileri şeklinde biyolojik veriler bulunduran ve genel kullanıma açık olan veritabanları ortaya çıktı. Tezin giriş kısmında anlatılan veri madenciliği yöntemi bu biyolojik veritabanlarından bilgi çıkarma düşüncesiyle artan bir biçimde bu veritabanları üzerinde kullanılmaya başlandı. Fakat biyolojik veritabanları alışveriş sepeti veritabanları gibi olmadığından, bu veritabanlarında uygulanacak veri madenciliği metodunun da alışılagelmiş klasik halinden farklı olması öngörülen bir durumdu. Zhang ve Ester'in [18] klasik veri madenciliği anlayışında olan aşağıdan-yukarı işleyiş yerine önerdiği yukarıdan-aşağı madencilik yöntemi kısaca bu kısımda anlatılacaktır.

2.2. Yukarıdan-Aşağı Tasarım ve Aşağıdan-Yukarı Tasarım Kavramları

Wikipedia ansiklopedisi yukarıdan-aşağı tasarımı sistemin genel bir bilgisinin sistem içerisindeki parçaların detayları verilmeksizin tanımlanması olarak anlatır. Sistem içerisindeki parçalar sonradan daha detaylı bir biçimde incelenerek tanımlamaları yapılır. Sonradan eklenen yeni parçalar da yeterli detaya sahip tanımlamaları yapıldıktan sonra modele entegre edilir. Yukarıdan-aşağı model çoğu zaman "kara kutular" yardımıyla tasarım yapmaktır. Gerçeklemesi kolay olmakla beraber temel mekanizmaların anlaşılmasını mümkün kılmaz.

Farklı olarak aşağıdan-yukarı tasarımda sistemin her bir parçası detaylı bir biçimde tanımlanır. Bu parçalar birleştirilerek daha büyük kısımlar oluşturulur. Bu daha büyük kısımlarsa bütün system ortaya çıkana dek birleştirilmeye devam edilir.

Yukarıdan-aşağı yaklaşımlar genellikle buluşlara ve gözlemlere dayanır, buna karşın aşağıdan-yukarı yaklaşımlar temel prensipler üzerinden yola çıkar [19].

2.3. Yukarıdan-Aşağı Veri Madenciliği Kullanımının Nedenleri

Biyolojik veritabanlarında veri madenciliğinin en yaygın kullanım şekli sık tekrarlanan örüntülerin bulunmasıdır. Başka bir ifadeyle, tekrar sayısı belirlenmiş bir destek değerinin üzerinde olan örüntülerin bulunmasıdır. Bu tip bir veri madenciliğinin yapılmasını sağlayan

unsurlar şunlardır: benzer diziler yüksek bir ihtimalle benzer fonksiyonlar gösterir ve biyolojik verilerin büyük bölümünde genel olarak gürültü bulunur.

Bu örüntüleri bulmak için uygulanan strateji genelde klasik apriori anlayışı olmuştur, yani aşağıdan yukarı giden bir biçimde veri madenciliği yapılmıştır. Bu anlayışta önce kısa uzunluktaki sık tekrarlanan örüntüler bulunur, kısa uzunlukla başlanarak bir anlamda aşağıdan başlanılıp yukarı doğru daha uzun sık tekrarlanan örüntülere gidilir. Bu anlayışın biyolojik veritabanlarında başarısını engelleyen iki büyük zayıflığı bulunmaktadır [18]:

i-) Bu anlayışta algoritmanın bütün sık tekrarlanan örüntüleri üreterek çalışmaya başlaması, biyolojik veritabanlarında bulunan verilerin karakteristikleri yüzünden başlangıçta çok fazla sayıda örüntü üretilmesine neden olur.

ii-) Bu sık tekrarlanan örüntüler uzunluklarının kısa olması nedeniyle biyolojik açıdan çok anlamlı değildir.

Bu zayıflıkların üstesinden gelmek için önerilen yöntem yukarıdan-aşağı veri madenciliğidir. Arama işlemi daha genel ve daha uzun bir örüntü aranmasıyla başlar. Bu örüntü sık tekrarlanıyorsa genişletilip daha uzun ve sık tekrarlanan bir örüntü aranır. Çok-değerli destek değeri kullanmayan yaklaşımlarda sık tekrarlanan en uzun örüntü bulunduktan sonra onun daha kısa parçaları da sık tekrarlı örüntü olur.

2.4. Yukarıdan-Aşağı Veri Madenciliğinin Dezavantajı

Yukarıdan-aşağı veri madenciliği metodu sunduğu avantajların yanısıra birde dezavantaja sahiptir. Maksimum uzunluğun kullanıcı tarafından veriliyor olması istenilen bir durum değildir. Çünkü kullanıcı veri madenciliği işlemini kendisinin bilmediği saklı bilgiyi çıkarmak için yapacağından böyle bir bilgiye önceden sahip olması zor bir durum olarak görülebilir.

Yukarıdan-aşağı veri madenciliğinin, önerilen yöntemde kullanılması bahsedilen bu dezavantajın yaşanmasına neden olmaz. Motif arama probleminde bulunması istenilen motiflerin maksimum uzunluğu öngörülür bir bilgi olduğu için bu yönteme ait bu dezavantaj herhangi bir olumsuz durum yaratmamaktadır [1].

3. MOTİF BULMA PROBLEMİ VE MEVCUT MOTİF BULMA YÖNTEMLERİ

Motif bulma problemi, giriş olarak verilen sıralar üzerinde farklı noktalarda bulunan ve önceden ne olduğu bilinmeyen bir örüntünün bulunması problemi olarak tanımlanabilir. Biyolojik açıdan önemi giriş kısmında anlatılmış olan bu problemi çözebilmek için bir çok çalışma yapılmıştır ve halen yapılmaya devam etmektedir. Bu kısımda bu çalışmalardan popüler olan birkaçı verilecektir.

3.1. MEME (Multiple EM for Motif Elicitation)

MEME [20] çok bilinen ve başarılı bir motif arama algoritmasıdır. Timothey Bailey tarafından doktora tezi için 1995 yılında geliştirilmiştir. Bu algoritmanın işleyişi kısaca şu şekilde anlatılır: Bayesian olasılıkları bir Expectation Maximization (EM) algoritması tarafından kullanılır ve kontrolsüz öğrenme hizalanmamış, boşluk bırakma işlemi yapılmamış bir dizi grubunda saklanmış motifi tespit etmede kullanılır. EM veri içerisindeki maximum benzerliği bulmada kullanılan genel bir metoddur [21].

MEME'nin diğer birçok motif bulma algoritmasından ayrılan yönü hizalanmamış motifleri de çıkarabilmesidir. Bu avantajına karşın MEME'nin orjinal amacının protein dizilerindeki motifleri çıkarma olması onu DNA verisi için tasarlanmış algoritmalar karşısında daha başarısız deneysel sonuçlar vermesine neden olur.

3.2. Gibbs Sampler

İlk olarak 1993 yılında Science dergisinde Lawrence ve diğerleri tarafından yazılmış bir makalede yer almış ve sadece "Gibbs" olarak adlandırılmıştır [22]. Bu tanımlamada yer alan varsayımlarından birisi her dizide en az bir motifin var olması olduğundan bu metod yer örnekleyicisi (site sampler) olarak da çağrılmıştır.

Gibbs, Markov zinciri ve Monte Carlo yaklaşımı kullanan bir algoritmadır. Markov zinciri kullanma nedeni her bir adım sonrasındaki sonuçların bir önceki adıma EM metodunda olduğu gibi bağımlı olmasıdır. Monte Carlo yaklaşımını ise bir sonraki adımı seçme yolunun deterministik değil rassal sayılar gibi örnekleme tabanlı olmasıdır. MEME'den göze çarpan tek farklılığı, EM metodunun beklenen değeri maksimize etmek için tek bir örnek seçmesine karşın Gibbs'de her bir örneğin seçilme işlemi için belli bir olasılığa sahip olmasıdır [21].

3.3. AlignACE (Aligns Nucleic Acid Conserved Elements)

Gerçek biyolojik örneklere daha iyi uyacak biçimde geliştirilmiş bir motif örnekleme algoritmasıdır [23]. DNA sıralarında saklı dizileri aramaya yarar. Harvard Medical School'da moleküler biyoloji araştırmacıları tarafından geliştirilmiştir. Bu yöntem Gibbs Sampler stratejisi uyguluyor olsa da klasik MAP-score ve Gibbs Sampler yöntemlerinden farklı olarak motifleri biyolojik olarak daha anlamlı değerlendiren bir yapıya sahiptir. AlignACE'le beraber ScanACE ve ComparaACE gibi araçlar da mevcuttur.

3.4. ANN-Spec

Stormo ve Workman tarafından ortaya atılan ANN-Spec yaklaşımı nöronlarla öğrenme konusundan yola çıkılarak yapılmış bir çalışmadır [24]. Bu yöntem bağlanma noktalarının tespiti için yapay sinir ağı kullanır. Gibbs Sampler yöntemine sinir ağı içerisindeki nöronların ağırlıklarının matris içerisindeki ağırlıklara karşılık gelmesi açısından benzerlik gösterir. Bu yöntemin Gibbs Samler ve MEME yöntemlerinden üstün olduğu iddia edilir [21].

3.5. CONSENSUS

Consensus, Hertz ve diğ. tarafından geliştirilen DNA veya protein sıraları üzerinde çalışabilen bir matris tabanlı motif arama yöntemidir [25]. Program başlangıçta 4 tane en anlamlı matrisi getirir. Bu matrisler ufak farklılıklarla beraber aynı motifi içerirler. Beklenen frekansları hesaplayarak yöntem motif arama işlemi gerçekleştirir.

4. MOTİF BULMA PROBLEMİNE ÖNERİLEN YÖNTEMLE YENİ BİR ÇÖZÜM

Bir önceki kısımda da bahsedildiği gibi motif bulma problemini çözmek için bir çok çalışma yapılmıştır. Bu çalışmalar arasında en popüler olanları olarak The Multiple Em for Motif Elicitation (MEME) [20], Gibbs sampler [22] ve CONSENSUS [25] verilebilir. Diğer algoritmalar genellikle motif arama araçlarını performans ve motif uzunluğu gibi kıstaslar bakımından geliştirmek için ortaya çıkmıştır. Stine, Dasgupta and Mukatari genetik algoritmayı Structured Genetic Algorithm (St-GA) adını verdikleri algoritmalarında kullanarak gen dizilerinde motif arama yaptırmışlardır [26], Liu ve diğ. de gene genetik algoritma kullanarak transcription başlangıç yerlerindeki (promoterlar) potansiyel motifleri bulmaya çalışmışlardır [27], Pan ve diğ. ise MacosFSpan ve MacosVSpan algoritmalarını biyolojik verilerin içerdiği maksimum uzunluktaki en sık tekrarlanan dizileri bulmada kullanmışlardır [28].

MacosFSpan ve MacosVSpan apriori tarzı algoritmaların etkisizliğini vurgulayıp biyolojik dizilerde daha iyi çalışan bir veri madenciliği çözümü ararken [28], Liu ve diğ. [27] genetik algoritma yaklaşımını çoklu dizi hizalama araçlarıyla birleştirerek motif bulma problemini çözmeye çalışır. St-GA algoritması da [26] benzer bir biçimde çalışır ve motif arayabilmek için çoklu dizi hizalama yapmaya ihtiyaç duyar.

Bu tezde yapılan çalışmanın amacı motif bulma problemine yeni bir çözüm getirilmesidir. Önerilen bu çözüm hem veri madenciliği hem de genetik algoritmadan faydalanır. Diğer yöntemlerde kullanılan çoklu dizi hizalama araçlarının kullanılmaması çalışma zamanı açısından ciddi bir kazanç sağlar. Bu performans zaafiyeti zamansal maliyeti yüksek olan BLAST [29] çoklu dizi hizalama algoritması kullanan St-GA algoritmasında da [26] özellikle vurgulanılmıştır.

4.1. Önerilen Yöntem

Motif probleminin çözümü için bu tezde önerilen yöntem yukarıdan-aşağı veri madenciliği ve genetik algoritma tabanlı hibrit bir çözümdür. Önerdiğimiz çözüm biyolojik dizi veritabanlarındaki motiflerin keşfinde kullanılır. Bu yöntemdeki yaklaşım iki temel adımda ele alınabilir. Birinci adım, genetik algoritma kullanılarak aday motiflerin bir popülasyonun oluşturulmasıdır, bunu diğer nesillerin genetik operatörler ve uygunluk fonksiyonu kullanılarak oluşturulması takip eder. İkinci adımda, veri madenciliği yöntemi yukarıdan-aşağı haliyle kullanılarak aday motiflerin uygunluğunun değerlendirilmesi yapılır. Bu sayede dizi hizalama

araçlarını kullanmaya gerek kalmadan sadece veri madenciliği kullanılarak potansiyel motiflerin ve tekrarlı örüntülerin veriden çıkartılması sağlanır. Önerilen bu yöntemde, arama uzayını daha da genişletmek ve daha iyi sonuç alabilmek için birçok motif arama algoritmasından farklı olarak belirsizlik eklentisi de bulunmaktadır.

4.1.1. Veri Madenciliği Tanımlamaları

Σ bir semboller kümesini gösteriyor olsun, *alfabe* olarak da adlandırılır. Alfabe DNA sıralarından oluşan veri kümeleri için Σ = {A, T, C, G} şeklinde tanımlanır, bu harfler birinci bölümde anlatılan DNA sıralarındaki bazlara karşılık gelmektedir. Veri kümesi D, n tane sıradan oluşuyor olsun, her bir sıra için $s = s_1s_2...s_k$, $s_j \in \Sigma$ ve $1 \leq j \leq k$ tanımlamalarını yapalım, k da sıra uzunluğunu göstersin.

Bir *örüntü* $p = a_1a_2...a_k$ şeklinde bir sıradır, burada $a_i \in \Sigma$'dır. k ise örüntünün uzunluğudur, |p| olarak da gösterilebilir.

İki örüntümüz olsun $p = a_1a_2...a_j$ ve $p' = a'_1a'_2...a'_i$ şeklinde. Eğer $\exists$ a için, $1 \leq x \leq i : \forall$ y, $1 \leq y \leq j : a'_{x+y-1} = a_y$ şartı sağlanıyorsa p örüntüsü, p' örüntüsünün *altsırası* olarak adlandırılır.

$p' = a'_1a'_2...a'_i$ bir örüntü ve $n < |p'|$ olmak üzere, bir *n-altsıra kümesi* p' örüntüsünün bütün n uzunluktaki altsıralarını içeren küme olarak tanımlanır. Örneğin, ATCT örüntüsünün 3-altsıra kümesi {ATC, TCT} şeklindedir.

Lemma: p', k uzunluğunda bir örüntü olsun. k > m için, p' örüntüsünün (k-m)-altsıra kümesi m+1 elemandan oluşur.

Veri madenciliği kullanılarak sadece örüntülerin ve onlara ait altsıraların destek değerleri kontrol edilir. Bu durum katı bir kesinliğe götürmektedir. Ancak, içinde gürültü barındırabilen biyolojik veriler üzerinde çalışırken esneklik de önemli bir konudur. Bu nedenle IUPAC bir-harf belirsizlik kodları [30] belirsiz DNA bazlarının tespit edilmesi için yönteme dahil edilmiştir. Bu kodlar DNA sıralarında yer alan belirsiz bazları saptamaya yarar. Standart olarak kabul edilmiş bu kodlardan ikili olanları sırasıyla M = A veya C, R = A veya G, W = A veya T, S = C veya G, Y = C veya T, K = G veya T şeklindedir.

Bir $p = e_1e_2...e_k$ örüntüsünün *belirsiz örüntü*sü $1 \leq i \leq k$ olmak üzere sadece ve sadece bir $e_i \in$ {M, R, W, S, Y, K} içeren örüntü olarak tanımlanır.

4.1.2. Kullanılan Uygunluk Fonksiyonu ve Genetik Operatörler

Uygunluk fonksiyonunu hesaplanırken, üç tane faktöre dikkat edilmesi gerekir: örüntü uzunluğu, destek değeri ve belirsizliğin olup olmaması. Ayrıca uygunluk fonksiyonunun aşağıdaki şartları sağlaması da gerekir:

i) Aday örüntülerden daha uzun olanları daha kısa olanlarından daha yüksek bir uygunluk değeri almalıdır.
ii) Aday örüntülerden birbirlerine eşit uzunluklarda olanlardan daha yüksek destek değerine sahip olan daha yüksek bir uygunluk değeri almalıdır.
iii) Aday örüntülerden birbirlerine eşit uzunlukta olanlardan belirsizlik içerenin daha düşük bir uygunluk değeri alması gerekmektedir.

Sonuç olarak kullanılan ağırlıklandırılmış uygunluk fonksiyonu aşağıdaki gibi olmuştur:

$$F(P_i) = w_1 * |C| + w_2 * \delta_c - w_3 * \alpha$$
$$w_1 + w_2 - w_3 = 1$$

Fonksiyondaki i örüntünün index değerini gösterir, $|C|$ örüntünün uzunluğudur, δ_c destek değerini gösterir, ve α, belirsizlik olduğu zaman 1 değerine eşittir, diğer durumdaysa 0 değerini alır. Ağırlık değerleri de yukarıda verilen şartları yerine getirebilir bir şekilde ayarlanmalıdır.

Genetik Algoritmaların işleyişini sağlayan bir diğer unsur genetik operatörlerdir. Bir sonraki nesil hazırlanırken temel genetik operatörler arasında yer alan mutasyon ve çaprazlama operatörleri çeşitlilik yaratma amacıyla kullanılmışlardır.

Mutasyonlar iki durumda kullanılır. İlki popülasyona ait toplam uygunluk değerinin bir önceki popülasyona göre daha düşük değer aldığı durumdur. Bu durumda rasgele bir mutasyon rasgele seçilmiş bireylere uygulanır. İkincisi ise popülasyon havuzunda aynı üyelerin bulunduğu durumdur. Bu durumdaysa iki bireyden birine mutasyon uygulanır.

Çaprazlama operatörü yeni bireyler oluşturulurken kullanılır. Tek-nokta çaprazlama operatörünün kullanılması tercih edilmiştir.

4.1.3. Algoritma

Şekil 4.1 de önerilen yöntemin kullandığı algoritma prosedürel bir yapıda sunulmuştur. Bunu takip eden Şekil 4.2. deyse algoritmanın ihtiyaç duyduğu yukarıdan aşağı veri madenciliği işleminin algoritması bulunmaktadır.

Minimum destek değerini belirle: δ
Motif uzunluğunu belirle: max-length
Minimum altsıra uzunluğunu belirle: min-length
Toplam iterasyon sayısını belirle: i
Rassal olarak aday motifleri yarat: $C_1 \sim C_m$
Veri kümesi D'den arama yapılacak sıraları getir: $S_1 \sim S_n$

while iterasyon değeri $\leq$ i **do**
    for her bir aday motif C_m **do**
        Uygunluk değerini Yukarıdan-Aşağı-Veri-Madenciliği ile hesapla
    end-for
    Daha yüksek uygunluk fonksiyonu değerine sahip aday motifleri seç ve genetik algoritmaları kullanarak sonraki nesili hazırla.
    İterasyon değerini 1 arttır
end-while

Bulunmuş motifleri çıktı olarak ver.

Şekil 4.1: Önerilen yöntemin algoritması

```
Yukarıdan-Aşağı-Veri-Madenciliği(veri kümesi D, minimum_destek δ, aday motif C,
minimum_uzunluk min-length)
m = |C|
while m ≥ min-length do
        C'ye ait m-altsıra kümesini bul
        for C'nin m-altsıra kümesindeki her bir eleman için do
                destek değeri δc'yi bul
                if δc ≥ δ
                        C'nin uygunluk değerini güncelle, for döngüsünü bitir ve return et
                else elemana ait bütün belirsiz örüntülerin destek değerlerini kontrol edip daha
                yüksek bir uygunluk değeri ara
        end-for
        m değerini 1 azalt
end-while
return C'ye ait uygunluk değeri
```

Şekil 4.2. Yöntemde kullanılan yukarıdan-aşağı veri madenciliği algoritması

4.2. Deneysel Sonuçlar

4.2.1. Kullanılan Veri Kümesi

Kullanılan veri kümesi E.coli bakterisinin DNA'sından alınan ait 300 tane promoter sırasıdır. Her bir sıra 100 baz uzunluğuna sahiptir. Bu veri kümesi daha önce Horton'un çalışmasında da [31] kullanılmıştır, ve bu veri kümesine internet aracılığıyla erişebilmek mümkündür [32].

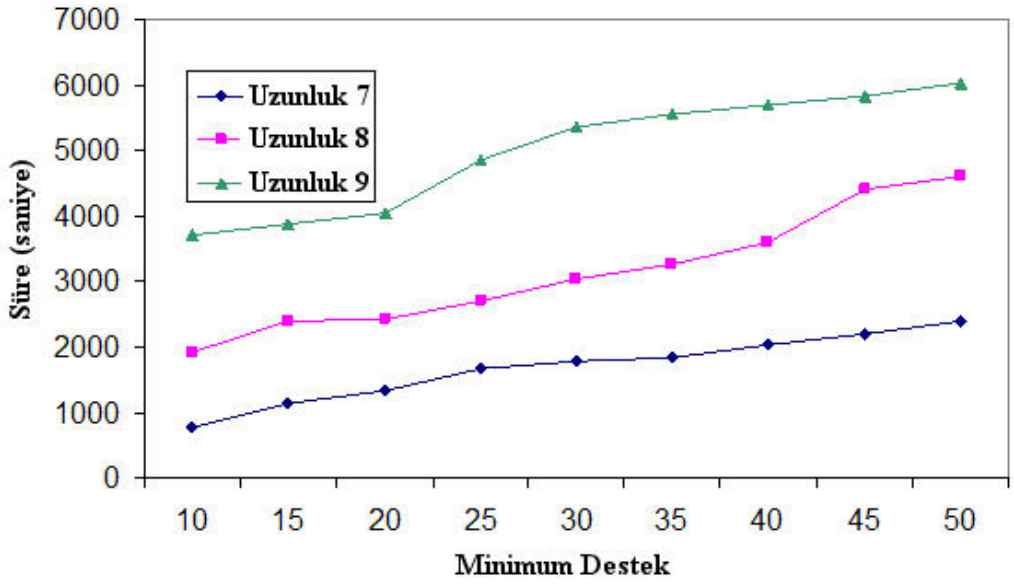
4.2.2. Gerçekleme ve Test Ortamı

Önerilen yöntemin gerçeklenirken JAVA programlama dili kullanılmıştır. Saf Java bileşenleri seçilerek platform bağımsızlığı özelliği korunmaya çalışılmış, ayrıca CPU ve hafıza kullanımını azaltmak için kod üzerinde bazı optimizasyonlar da yapılmıştır.

Testler AMD 3000+ işlemciye ve 1.5 Gigabyte DDR-400 hafızaya sahip bir PC üzerinde yapıldı. Yazılım tarafındaysa, en son güncellemeleri yapılmış olan Microsoft Windows XP ve Microsoft SQL Server 2000 koşturulmuştur.

4.2.3. Sonuçlar

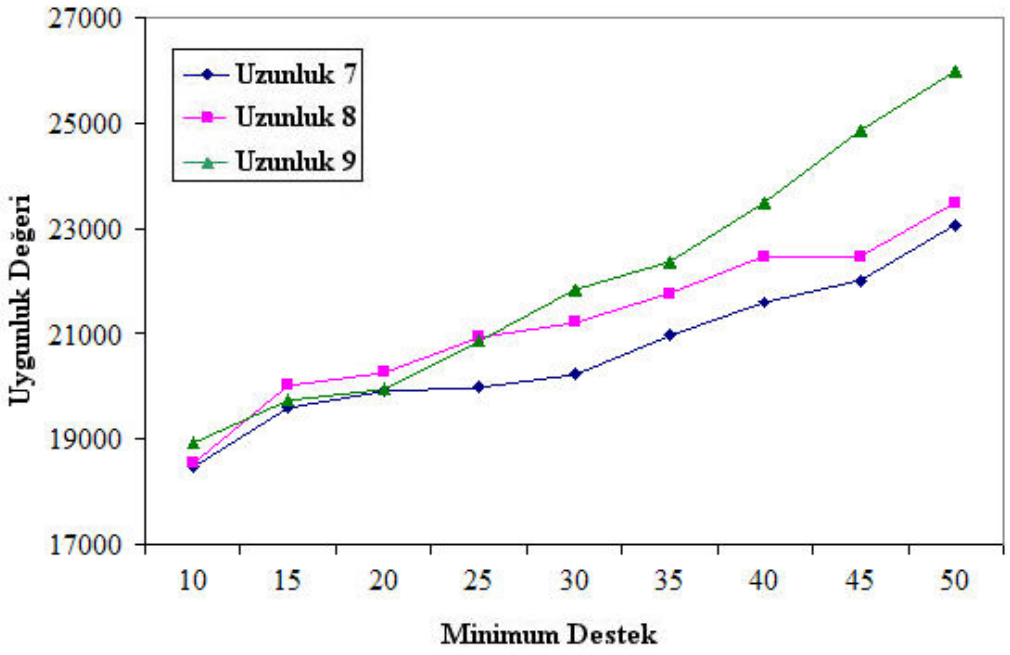
Şekil 4.3-6. da elde edilen test sonuçları gösterilmekte. Şekil 4.3. ve 4.4. de minimum destek değerinin değişmesiyle elde edilen uygunluk değeri ve performans verileri farklı motif uzunlukları üzerinde incelenilmiştir. Şekil 4.5. de ise önerilen yöntem farklı yöntemlerle kıyaslanarak performans üstünlüğüne bakılmıştır. Son olarak Şekil 4.6 da sürenin örüntü uzunluğuna bağlı değişim grafiği verilerek önerilen yöntemin artan uzunluklarda da performanslı olduğu gösterilmiştir. Bu sonuçlarda da görüleceği gibi önerilen yöntem performans açısından beklenildiği gibi iyi bir sonuç ortaya koymaktadır.



Şekil 4.3. Farklı minimum destek değerleri için çalışma süreleri (saniye)

Şekil 4.3. de yapılan testlerde algoritma üzerinde aranan örüntü uzunlukları ve uzunluğa ait farklı minimum destek değerleri kullanılarak bir karşılaştırma yapılmıştır. Uzunluk arttıkça geçen sürenin artmakta olması beklenilen bir durumdur. Minimum destek değerinin artmasıyla geçen sürenin artmasının nedeni yukarıdan aşağı veri madenciliğidir. Madencilik işlemi esnasında daha uzun ve desteği az olan potansiyel örüntüler minimum destek değeri arttıkça bu desteği sağlayamayacaktır. Bu da algoritmanın yukarıdan bir birim aşağıya inmesine ve daha fazla miktarda arama yapmasına neden olur, daha fazla arama da zamanda artışın nedenidir. Örnekle açıklarsak ATGC şeklindeki bir aday örüntü destek değerini sağlamazsa algoritma bu örüntünün belirsizlik hallerini ve daha sonra altsıralarını incelemeye alır. Bu durum da zamansal bir artışı beraberinde getirir.

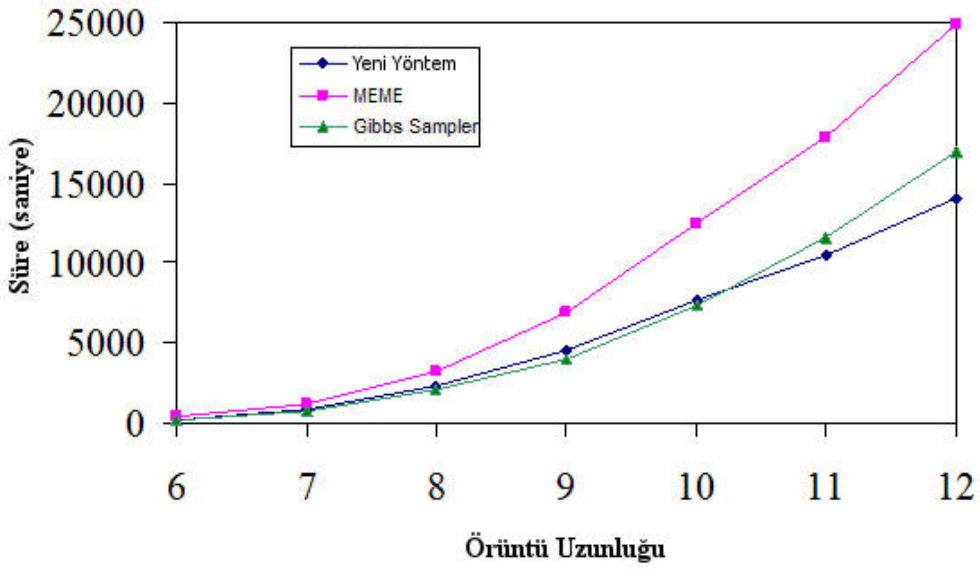
Burada göze çarpan bir diğer sonuç farklı uzunluklar için sergilenen karakterin benzerliğidir. Bu durum algoritmanın ölçeklenebilir olduğuna işaret eder.



Şekil 4.4. Farklı minimum destek değerleri için uygunluk değeri karşılaştırması

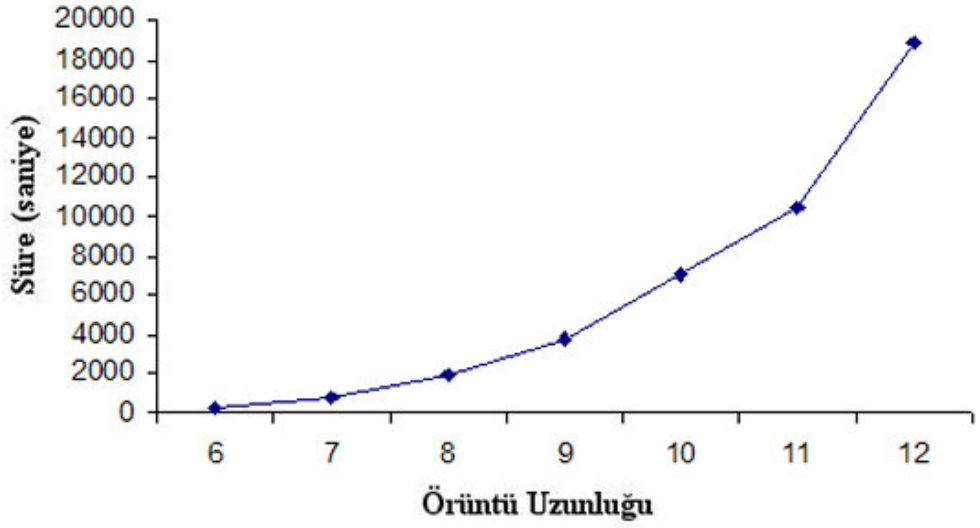
Şekil 4.4. de farklı uzunluklar ve onlara ait farklı minimum destek değerleri ile edilen motiflerin uygunluk değerleri karşılaştırılmıştır. Minimum destek değerinin artmasıyla uygunluk değerinin artıyor olması beklenen bir durumdur çünkü destek değeri arttıkça bulunan motiflerin kalitesi de artar. Bir önceki deney sonuçları hatırlanırsa bu kalite artışının beraberinde zamansal bir artışı da getirdiği görülür.

Grafiklerde düşük destek değerleri için kesişme noktalarının olması düşük destek değerlerinde ayırt ediciliğin az olması, başka bir ifadeyle bir miktar gürültünün karışması olarak düşünülebilir.



Şekil 4.5. Farklı yöntemlerin performans (süre) açısından karşılaştırılması

Şekil 4.5. de önerdiğimiz yöntemin diğer bazı motif arama yöntemleriyle örüntü uzunluğunun zamana bağlı artışı açısından karşılaştırılması yapılmıştır. Gibbs Sampler [22] algoritmasının performans açısından MEME'den [20] daha iyi sonuç verdiği fakat üretilen motiflerin kalitesi açısından daha kötü olduğu söylenmektedir. Bu grafikte de performans açısından bu durum gözükür. Önerilen yöntemin performansı öngörülen biçimde yüksek çıkmıştır.



Şekil 4.6. Örüntü uzunluğu - süre grafiği

Son olarak Şekil 4.6. da örüntü uzunluğu – süre grafiği verilmiştir. Örüntü uzunluğu arttıkça sürenin artması beklenilen bir durumdur. Dikkat edilirse 11 uzunluğundaki örüntüye kadar bu artış doğrusala yakın bir durumdadır. Bu da önerilen yöntemin performans açısından stabil olduğunu ve yüksek örüntü uzunluklarında bile iyi performans sergilediğini gösterir. Çoğu motif arama algoritması 9 uzunluğundaki bir motif arama işlemini yapamamakta ya da yaparken kabul edilebilir sınırın üzerinde zaman harcamaktadır.

11 uzunluğundaki örüntüden sonraki daha yüksek artışın arama sayısının artmasının yanısıra işlemci ve hafıza gibi donanımsal nedenlere bağlı olması da olası bir durumdur. Çünkü motif uzunluğunun artması sonucu daha fazla yer tutan bir veri üzerinde işlem yapılmasına neden olacaktır. Bu durumsa mevcut yapıda donanım yetersizliğine neden olabilir.

5. GELİŞTİRİLEN UYGULAMA

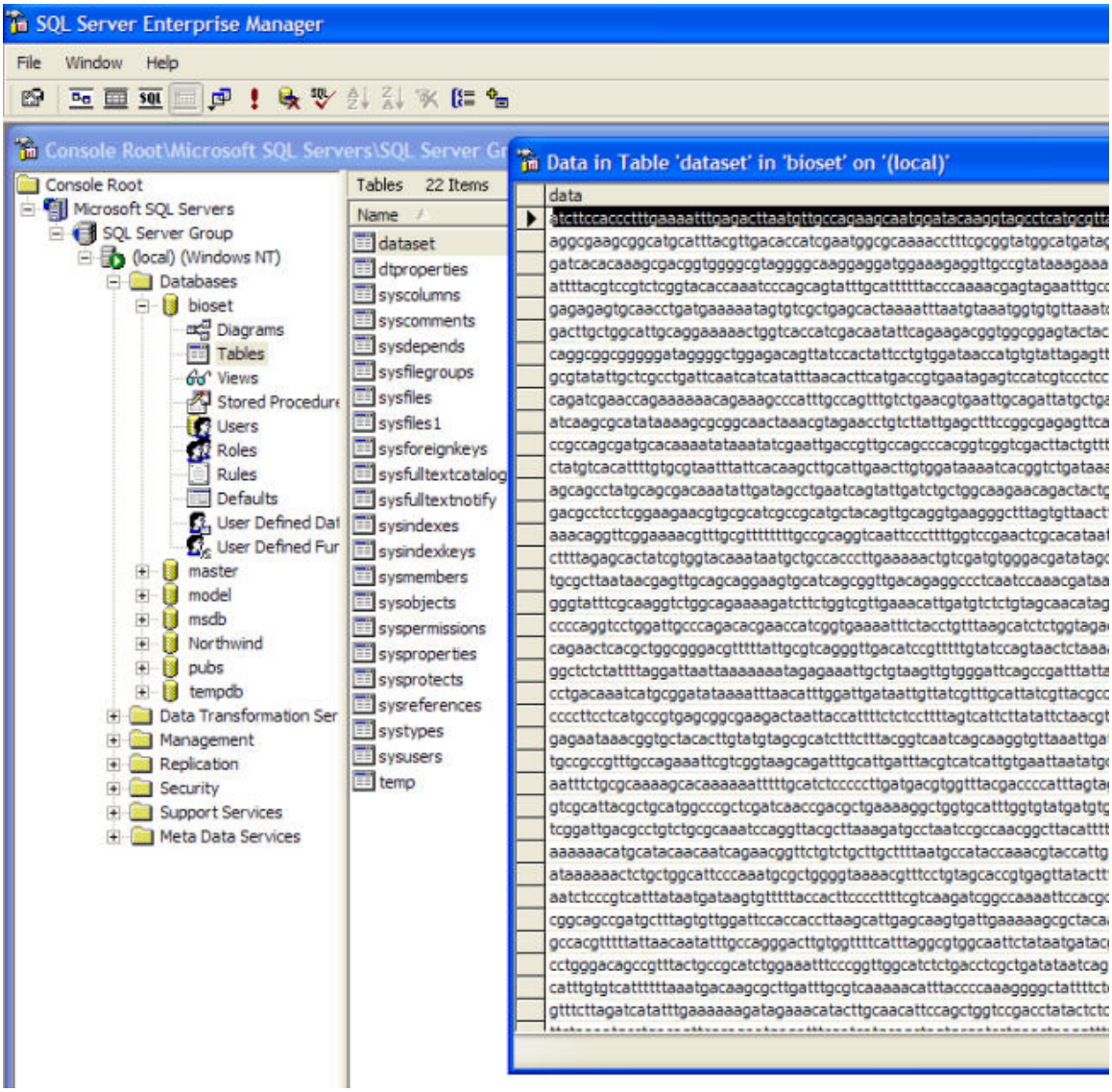
5.1. Giriş

Bir önceki kısımda anlatılan yöntemin çalışmasını test etmek ve bu tez çalışmasının amaçlarından birini gerçekleştirmek için Top-Down GA ismini verdiğimiz bir yazılım geliştirildi. Bu kısımda bu yazılım tanıtılacak ve nasıl kullanıldığı anlatılacaktır.

Bu yazılım Java programlama dili kullanılarak geliştirilmiştir. Java dilinin seçimindeki temel neden ileride bu yazılımın dağıtık bir yapıda çalışır hale getirilmesini ve farklı işletim sistemi kullanan istemciler üzerinden (cross-platform) de çalışmasını sağlayabilmektir. Program geliştirmesi esnasında Borland JBuilder 2005 ve JDK 1.4. kullanılmıştır.

Önerilen yöntem DNA sıralarından motif çıkarma ve potansiyel örüntülerin keşfi işlemlerini yapmaktadır. Bu nedenle ihtiyaç duyulan veritabanına, program JDBC:ODBC köprüsü üzerinden erişilen Microsoft SQL Server 2000 sunucusuyla bağlanır. Program çalıştırılmadan önce ODBC ayarları ve veritabanı ayarlarının yapılması gerekir.

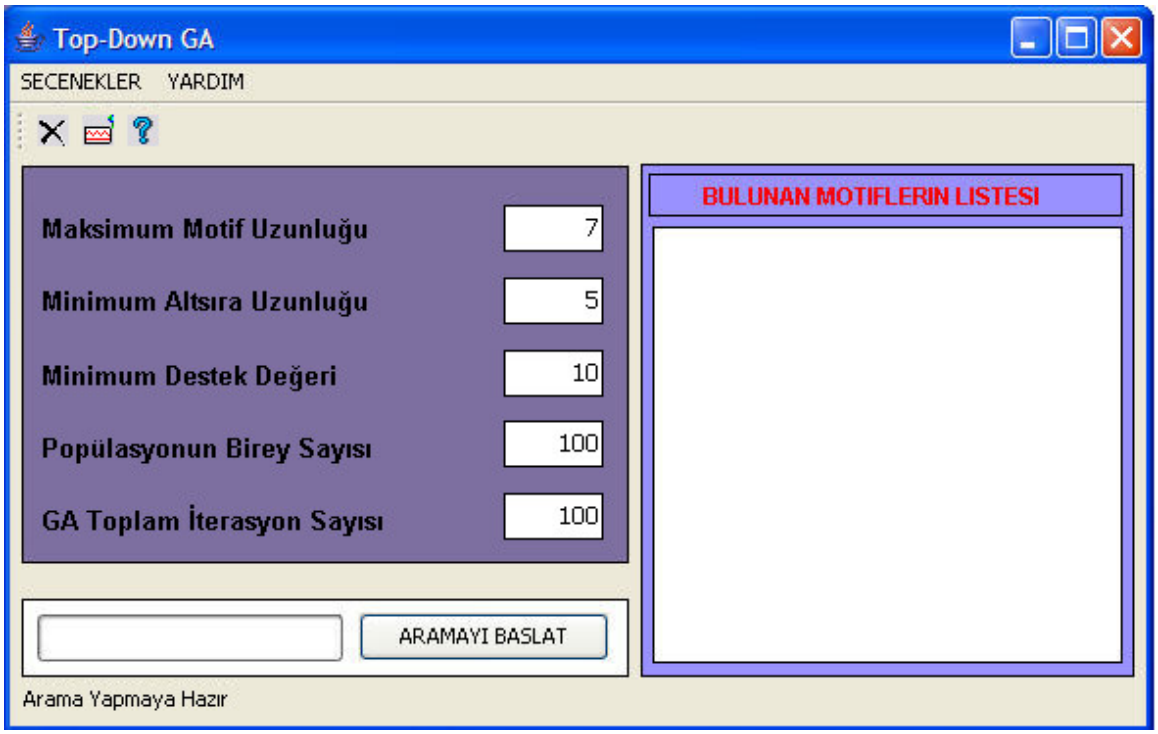
Bir önceki kısımda verilen test sonuçları alınırken E. coli bakterilerine ait DNA promoter sıralarından oluşan bir veritabanı kullanılmıştı. Şekil 5.1. de bu veritabanından bir görünüm verilmekte.



Şekil 5.1. Microsoft SQL Server 2000 üzerinde tutulan E. coli DNA promoter sıraları.

5.2. Hazırlanılan Uygulama

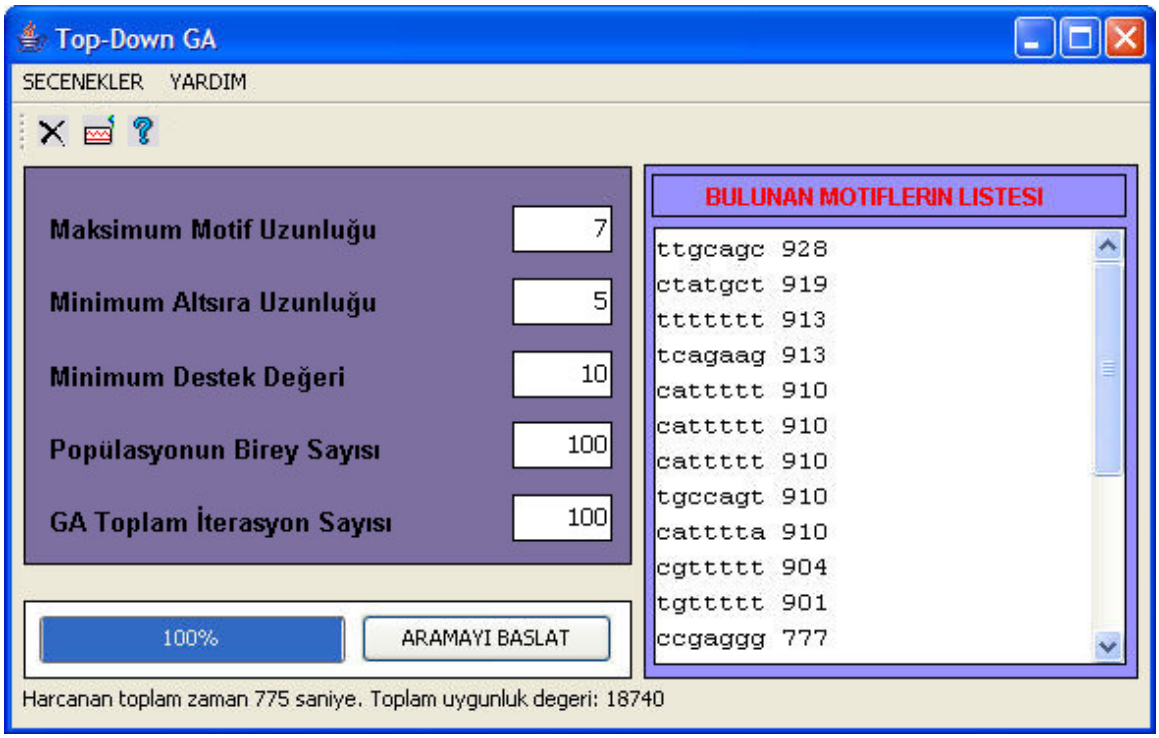
Kullandığı yöntem açısından hem yukarıdan-aşağı veri madenciliği hem de genetik algoritma barındırdığı için bu programı Top-Down GA olarak adlandırıldı. Programa ait ekran çıktısı Şekil 5.2. de görülmektedir.



Şekil 5.2. Top-Down GA Olarak Adlandırılan Uygulama

Görüldüğü gibi program kolay anlaşılır bir kullanıcı arayüzüne sahiptir. Sol kısımda önerilen yöntemde bulunan bazı parametrelerin kullanıcı tarafından girilebileceği bir alan mevcuttur. Bu kısımdan kullanıcı aramak istediği motif uzunluğunu, minimum altsıra uzunluğunu, minimum destek değerini, popülasyonun birey sayısını ve genetik algoritmanın toplam iterasyon sayısını girebilir. Şekil 5.2. de görünen ekranda program açıldığı zaman varsayılan atanmış değerler bulunmaktadır. Bu kısmın altında yer alan bölümde arama işlemini başlatmaya yarayan düğme ve onun sol tarafında arama durum göstergesi bulunmaktadır. Alt tarafta durum çubuğu, sağ tarafta da elde edilen sonuçların yer alacağı ekran yer almaktadır.

Programın çalışması tamamlandıktan sonra Şekil 5.3. deki gibi bir ekran çıkarak sonuçlar kullanıcıya ulaştırılır. Kullanıcı isteğe bağlı olarak bu ekrandan yeni bir arama başlatabilir ya da programı sonlandırabilir.



Şekil 5.3. Top-Down GA olarak adlandırılan uygulama çalışma işlemi bittikten sonra

6. SONUÇ

Bu tez çalışmasında birden çok alanı kapsayan bir çalışma yapılmıştır. Biyoenformatik ve veri madenciliği gibi son yılların popüler iki alanı bir araya getirilmiş ve bu birliktelik kullanılarak motif bulma problemine farklı bir çözüm metodu aranmıştır. Motif bulma problemi, sunduğu faydalar bakımından üzerinde çok durulan bir konudur. Bu konuda üretilen yeni yaklaşımlar daha iyi araçların üretilmesine olanak sağlayarak moleküler biyoloji ve tıp alanında çalışan araştırmacılara yardımcı olacaktır. Bunun da geri dönüşü günümüzde tedavisi mevcut olmayan kanser gibi bazı hastalıklara tedavi bulunması, daha az yan etkisi olan ya da yan etkisi hiç olmayan yeni ilaçların keşfi ve canlı türleri arasındaki akrabalık ilişkilerinin daha iyi anlaşılması gibi şekillerde olabilir.

Genetik kod olarak adlandırılan saklı bilgi, bildiğimiz üzere DNA moleküllerinde bulunmaktaydı. Genom projeleri sonucu insan da dahil olmak üzere bazı canlı türlerine ait bu bilgiler günümüzde gen veri bankalarına aktarılmış durumdadır. Bu veriler üzerinden bilgi çıkarmak için çeşitli yaklaşımlar denenmiştir. Yapılan tez çalışmasında biyolojik sıralardaki tekrarlı örüntülerin ve potansiyel motiflerin veri madenciliği yöntemiyle çıkarılması işlemi yapılırken mevcut yaklaşımlardan zaman açısından daha performanslı çalışan bir yöntem geliştirilmiştir. Mevcut yaklaşımlar çeşitli yan araçlarla genellikle çoklu dizi hizalaması yapmaktadırlar. Çoklu dizi hizalaması ise zamansal maliyeti yüksek olan bir algoritmadır. O nedenle bu yöntemi kullanan algoritmalar, kullandıkları sıraların sayısı arttıkça ya da aranan örüntünün uzunluğu arttıkça zaman açısından etkisiz bir davranış sergiler. MEME [20] gibi literatürde ismi çok fazla geçen bir motif arama algoritmasının uygulaması bu nedenden ötürü bir süper bilgisayar üzerinde çalıştırılmaktadır.

Bu sebepten ötürü yapılan çalışmada öncelikli olarak çoklu dizi hizalama yapmadan bir arama yönteminin geliştirilmesi üzerinde durulmuştur. Bunun da çözümü, genetik algoritma ile veri madenciliğini birleştirip kendi ihtiyacı doğrultusunda kullanan bir yöntemle bulunmuştur. Burada yaşanabilecek problemlerden bir tanesi klasik anlamda yapılan veri madenciliği yönteminin de biyolojik sıralar üzerinde etkili çalışmamasıydı. Karşılaşılan bu sorun da alışıldığın aksi yönde yukarıdan aşağıya madencilik yöntemi kullanılarak aşıldı.

Önerilen yöntem Java programlama diliyle yazılmış bir uygulama ile E. coli bakterilerinden alınmış DNA promoter sıralarında motif arama işlemi yaptırılarak test edildi. Test sonuçlarında yöntemin performans açısından öngörülen başarıyı sergilediği görüldü.

Burada yapılan çalışma elde edilen motiflerin biyolojik açıdan kalitesinin arttırılması yönünde geliştirmeye açık bir durumdadır. İleride kullanılan genetik algoritmanın uygunluk fonksiyonunun çok amaçlı bir uygunluk fonksiyonu ile değiştirilmesi de düşünülebilir. Yazılım mühendisliği bakımından algoritmayla gerçekleştirilen uygulamayı dağıtık çalışan bir yapıda hazırlayarak performans arttırımı sağlanması veya uygulamanın web servisi halinde dışarıdan kullanıma açılarak daha iyi bir erişebilirlik sağlanması ileride çalışılabilecek konulardır.

KAYNAKLAR

1. Baloglu, U.B. and Kaya, M., "Top-Down Motif Discovery in Biological Sequence Datasets by Genetic Algorithm" IEEE International Conference on Hybrid Information Technology (ICHIT 2006), Chejhu Island, South Korea, 9-11 November 2006.
2. Setubal, J., Meidanis, J., Introduction to Computational Molecular Biology, PWS Publishing Company, 1997.
3. Hunter, L., "Life and Its Molecules A Brief Introduction", *AI Magazine*, American Association for Artificial Intelligence, 2004.
4. Mitra, S., Acharya, T., Data Mining: Multimedia, Soft Computing and Bioinformatics, Wiley Interscience, 2003.
5. Krawetz, S.A. and Womble, D.D., Introduction to Bioinformatics: A Theoretical and Practical Approach, Humana Press, 2003.
6. Internet: Nacional Center for Biotechnology Information, NCBI Home Page, http://www.ncbi.nlm.nih.gov, Erişim Tarihi: Ağustos 2006.
7. Internet: EMBL Heidelberg, The European Molecular Biology Laboratory, http://www.embl-heidelberg.de, Erişim Tarihi: Ağustos 2006.
8. Internet: The GDB Human Genome Database, http://www.gdb.org, Erişim Tarihi: Ağustos 2006.
9. Internet: RCSB Protein Data Bank, http://www.rcsb.org, Erişim Tarihi: Ağustos 2006.
10. Han, J., Kamber, M., Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, 2000.
11. Roiger, R.J., Geatz M.W., Data Mining: A Tutorial-Based Primer, Addison-Wesley, Pearson Education, Inc., 2003.
12. Bergeron, B., Bioinformatics Computing, Prentice Hall PTR, 2002.
13. Zhang, C. and Zhang S., Association Rule Mining, Springer-Verlag, 2002.
14. Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs, Springer Series Artificial Intelligence, Springer-Verlag, 1992.
15. Mitchell M., An Introduction to Genetic Algorithms, The MIT Press Cambridge, Massachusetts London, England, 1999.
16. Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J., Rapp, B.A., and Wheeler, D.L. "GenBank", *Nucleic Acids Research*, 30 (1):365-370, 2002.

17. Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.C., Estreicher, A., Gasteiger, E., Martin, M.J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S., Schneider, M., "The SWISS-PROT protein knowledgebase and its supplement", *Nucleic Acids Research*, 31 (1): 365-370, 2003.
18. Ester, M. and Zhang, X. "A Top-Down Method for Mining Most Specific Frequent Patterns in Biological Sequence Data", Proc. SIAM Int. Conf. on Data Mining (SDM'2004) 2004.
19. Bergeron, B., Bioinformatics Computing, Prentice Hall PTR, 2002.
20. Bailey, T.L. and Elkan, C. "Fitting a mixture model by expectation maximization to discover motifs in biopolymers", *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, Menlo Park, California, pp. 28-36, 1994.
21. Haussler M., "Motif Discovery on Promoter Sequences," Universitat Potsdam Institut für Informatik and IRISA/INRIA Rennes, 2005.
22. Thompson, W., Rouchka E.C. and Lawrence C.E. "Gibbs Recursive Sampler: Finding transcription factor binding sites", *J. Nucleic Acids Research*, Vol.31, pp. 3580-3585, 2003.
23. Internet: Harvard University, AlignACE Homepage, http://atlas.med.harvard.edu/, Erişim Tarihi: Ağustos 2006.
24. Workman, C.T. and Stormo, G.D., "ANN-Spec: A Method for Discovering Transcription Factor Binding Sites With Improved Specifity", Pacific Symposium on Biocomputing, Vol.5, pp. 464-475, 2000.
25. Hertz, G.Z., Hartzell, G.W. and Stormo, G.D. "Identification of consensus patterns in unaligned DNA sequences known to be functionally related", *Bioinformatics,* Vol.6, pp. 81-92, 1990.
26. Stine, M., Dasgupta, D. and Mukatira, S. "Motif Discovery in Upstream Sequences of Coordinately Expressed Genes", The 2003 Congress on Evolutionary Computation, pp.1596-1603, 2003.
27. Liu, F.F.M et al. "FMGA: Finding Motifs by Genetic Algorithm", *Proceedings of the Fourth IEEE Symposium on Bioinformatics and Bioengineering*, pp.459-466, 2004.
28. Pan, J. et al. "Efficient Algorithms for Mining Maximal Frequent Concatenate Sequences in Biological Datasets", *Proceedings of the Fifth International Conference on Computer and Information Technology*, pp.98-104, 2005.

29. Altschul, S.F., Gish, W., Miller, W., Myers, E.W. & Lipman, D.J. "Basic local alignment search tool." J. Mol. Biol. 215:403-410, 1990.

30. M. Scherf, A. Klingenhoff, T. Werner, "Highly Specific Localization of Promoter Regions in Large Genomic Sequences by PromoterInspector: A Novel Context Analysis Approach," J. Mol. Biol. 297 (3), pp. 599-606, 2000.

31. Horton, P., "Tsukuba BB: A Branch and Bound Algorithm for Local Multiple Alignment of DNA and Protein Sequences", Journal of Computational Biology, 8(3):pp.283-303, 2001.

32. Internet: National Institute of Advanced Science and Technology (AIST), Computational Biology Research Center (CBRC), Sequence Analysis Team, Motif Discovery Datasets, http://seq.cbrc.jp/motifDiscoveryResources/index.html, Erişim Tarihi: Ağustos 2006.

ÖZGEÇMİŞ

Ulaş Baran BALOĞLU

baloglu@firat.edu.tr

Fırat Üniversitesi
Bilgisayar Mühendisliği Bölümü
23119, Elazığ

1996 yılında Elazığ Anadolu Lisesi'nden mezun olduktan sonra Bilkent Üniversitesi Bilgisayar Mühendisliği ve Enformatik bölümünü burslu olarak kazandım. 2002 yılında ara verdiğim üniversite eğitimine 2003 yılı sonunda Ortadoğu Teknik Üniversitesi'nde Enformatik Enstitüsü Bilişsel Bilimler yüksek lisans programıyla yeniden başladım. Bu sürenin öncesinde Başarı Telekom ve Datasel A.Ş. şirketlerinde stajer; Tunka Yazılım şirketinde de bilgisayar mühendisi olarak bir süre çalıştım. Yüksek lisans eğitimi nedeniyle bir süre ara vermiş olduğum iş hayatına 2004 yılında Biometri-CS ve ASELSAN bünyesinde yazılım mühendisi olarak çalışmaya başlayıp geri döndüm. 2005 yılı başında Fırat Üniversitesi Bilgisayar Mühendisliği Bölümü tarafından açılan araştırma görevliliği sınavını kazanmam sonrasında eğitim ve iş hayatımı Elazığ ilinde sürdürmeye başladım. Halen Fırat Ünivesitesi Bilgisayar Mühendisliği bölümünde araştırma görevlisi olarak görev yapmaktayım. Çok iyi düzeyde İngilizce ve temel seviyede Almanca bilgisine sahibim. Teknik anlamda ilgi alanlarım veri madenciliği, yazılım mühendisliği, yapay zeka ve biyoenformatik konularından oluşmaktadır.